A robust and efficient algorithm for determining molecular connectivities

Craig Melton

Supervisor: Deborah Crittenden

submitted in fulfilment of the requirements for

Master of Science in Chemistry

2018

School of Physical and Chemical Sciences

University of Canterbury

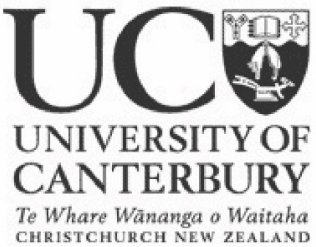


Table of contents

1. Introduction

1.1. Molecular dynamics simulations

Molecular dynamics (MD) simulations are a powerful tool for describing the structure, dynamics, and function of biomolecules in microscopic detail.[1] The role of numerical simulations in biochemistry has been steadily growing in recent decades.[2] The continuing growth in computer power[3-5] has made it possible to analyse, compare and characterise large and complex datasets obtained from computational experiments, like protein folding.[6-8] Proteins are particularly important biomolecules, because they perform important biological functions that are critical for sustaining complex life, such as biocatalysis[9-12], energy production, storage and processing[13], and electrochemical signal transduction[14-15] . Therefore, they have been the focus of most molecular dynamics studies.[16]

However, other biopolymers - DNA, RNA, polysaccharides - and biomolecules - sugars, lipids, neurotransmitters, nucleobases - are also important [17-20] but have received far less attention from the molecular simulation community.[21-22] This is largely due to the additional complexity associated with describing the structures and energies of these molecules and in particular interactions between them.[23-24]

1.2. Force fields

Underpinning all molecular dynamics simulations lie potential energy functions that describes how the potential energy within the molecular system changes as the atoms change position.[25-26] From the potential energy function, the forces acting on a given atom can be determined as the negative derivative of the potential energy function, U, with respect to Cartesian displacements: [27]

$$\vec{\mathrm{F}}(\vec{\mathrm{r}}) = -\nabla \mathrm{U}(\vec{\mathrm{r}}) = -\left\{\frac{\partial \mathrm{U}}{\partial \mathrm{x}}, \frac{\partial \mathrm{U}}{\partial \mathrm{y}}, \frac{\partial \mathrm{U}}{\partial \mathrm{z}}\right\}$$

where $\vec{\mathrm{r}} = \{\mathrm{x}, \mathrm{y}, \mathrm{z}\}$ is a vector specifying the position of the atom in Cartesian coordinates. The collection of all atomic position vectors will be denoted $\mathrm{R} = \mathrm{r_i}$. These atomic forces influence the dynamic behaviour of the system, which is simulated by numerically solving Newton's equations of motion as a function of time, for a series of small, discretized time steps.[28]

To progress further, it is necessary to specify an explicit form for the potential energy function, $\mathrm{U(R)}$, which depends on all of the coordinates of all of the atoms in the system.[25] Conventionally, the overall potential energy function is split into separate terms representing bonded and non-bonded interactions:[27]

$$\mathrm{U(R)} = \sum \mathrm{U_{bonded}\,(R)} + \sum \mathrm{U_{nonbonded}\,(R)}$$

Bonded interaction energies are conventionally separated into 3 distinct and decoupled components, arising from independent bond stretching, angle bending and torsional rotations:[27]

$$\mathrm{U_{bonded}(R)} = \mathrm{U_{str}(R)} + \mathrm{U_{bend}(R)} + \mathrm{U_{tors}(R)}$$

Pairwise covalent bonding interactions are typically described using a classical harmonic oscillator model:[27]

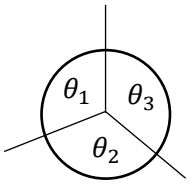
$$\mathrm{U_{str}(R)} = \sum_{\mathrm{ij}} \mathrm{k_{ij}(r_{ij} - r'_{ij})^2}$$

Where $\mathrm{r_{ij}} = \left\|\mathrm{r_i} - \mathrm{r_j}\right\|$ is the instantaneous distance between atoms *i* and *j*, and $\mathrm{r'_{ij}}$ represents the equilibrium bond length between atoms *i* and *j*, and $\mathrm{k_{ij}}$ the corresponding bond force constant. Both the $\mathrm{r'_{ij}}$ and $\mathrm{k_{ij}}$ parameters must be determined from either experimental[29-31] or computational [32-33] reference data. While other choices of potential energy function are available that are more accurate for large displacements from equilibrium, they require extra parameterization and so are not as widely used.[27, 34]

The most common functional form used to capture changes in energy upon angle bending is also harmonic: [27, 34]

$$\mathrm{U_{bend}(R)} = \sum_{\mathrm{ijk}} \mathrm{k_{ijk}(\theta_{ijk} - \theta'_{ijk})^2}$$

where $\mathrm{\theta_{ijk}}$ is the instantaneous value of the angle formed by connecting a central atom *j* to terminal atoms *i* and *k*, $\mathrm{k_{ijk}}$ is the corresponding angle bending force constant and $\mathrm{\theta'_{ijk}}$ the equilibrium angle. Once again, the $\mathrm{k_{ijk}}$ and parameters must be pre-fitted to appropriate reference data, and it is also necessary to define a single unique and physically meaningful set of angles, without double-counting or including angles that can otherwise be determined by symmetry or geometric constraints. An example of the angle redundancy problem is illustrated in Figure 1.



θ_3 = 360-θ_1-θ_2

Figure 1.1. Only two angles need to be specified to completely determine the conformation of a planar triatomic molecule, the third angle is given as the remainder required to complete the circle.

Energetic changes associated with bond rotation are generally described via truncated or single-term Fourier series expansion:[27, 34]

$$\mathrm{U_{tors}(R)} = \sum_{\mathrm{ijkl}} \mathrm{k_{ijkl}}\left(1 + \cos\left(\mathrm{n}\phi_{\mathrm{ijkl}} + \phi'_{\mathrm{ijkl}}\right)\right)$$

where *n* is the periodicity and is a non-negative natural number, ϕ_{ijkl} is the instantaneous value of the torsion angle, defined at the angle between the *i-j-k* plane and the *k-l* vector, and ϕ'_{ijkl} is a phase shift parameter that determines the minimum energy torsion angle. As in the angle bending case, it is necessary to define a unique and physically meaningful set of dihedral angles that describe all physically relevant bond rotations that only includes one dihedral angle per bond to be rotated. The associated phase shift and force constant parameters also need to be determined or specified.

Finally, it remains to define and compute non-bonded electrostatic and van der Waals interactions:[27]

$$\mathrm{U_{non-bonded}(R)} = \mathrm{U_{elec}(R)} + \mathrm{U_{vdW}(R)}$$

The non-bonding term represents all interactions between pairs of atoms that are not otherwise involved in bonding interactions. For clarity, these will be indexed by *m,n*, to denote their mutual exclusivity with the bonded atomic index sets.

Electrostatic interactions are most commonly modelled by computing classical electrostatic interactions between atom-centred point charges:[27, 34]

$$\mathrm{U_{elec}(R)} = \sum_{\mathrm{mn}} \frac{\mathrm{q_m q_n}}{\mathrm{r_{mn}}}$$

where *m* and *n* refer to atoms separated by more than 3 bonds, q_m and q_n are partial charges on atoms *m* and *n*, respectively and all quantities are expressed in atomic units. As for the parameters that occur in the bonded interaction terms, the partial charges must be determined *a priori* by fitting to reference data. These point charges generally parameterize the distribution of valence electrons within a molecular system.

The Lennard-Jones potential is most frequently used for describing van der Waals interactions between non-bonded atom-pairs:

$$U_{LJ}(R) = (-E_{min})\left[\left(\frac{r'_{mn}}{r_{mn}}\right)^{12} - 2\left(\frac{r'_{mn}}{r_{mn}}\right)^{6}\right]$$

where r_{mn} represents the instantaneous distance between non-bonded atom-pairs m and n, r'_{mn} is their preferred van der Waals contact distance and E_{min} is the depth of the potential well that arises from the counterbalance of weak long-range attractive induced-dipole induced-dipole interactions and strong short-range repulsive electronic and nuclear repulsion. The requisite parameters are often estimated by analogy to related rare gas atoms whose van der Waals potential energy functions can be measured experimentally.[27, 34] In some force fields, non-bonded interactions are also computed between torsionally-connected atoms, and the torsional parameters adjusted accordingly.[27, 34]

1.3. Molecular connectivities and topologies

The first step in any molecular dynamics simulation is to find a unique, non-redundant set of bond lengths, bond angles, dihedral angles and non-bonded atom pairs that determine which terms contribute to the potential energy function.[27] This is a non-trivial algorithmic and computational problem.[35-38] Common approaches used in existing molecular dynamics software are reviewed briefly below.

Amber, a large and widely-used program package for carrying out molecular dynamics simulations, has an internal program named LEaP.[34, 39-40] LEaP uses a rules-based algorithm to identify amino acid residues within proteins or nucleic acids within DNA/RNA from their specifiers within a PDB-formatted input file, and assign their connectivities and topologies.[34] However, it far from universally applicable, as it requires connectivity and non-redundant coordinate sets for unique non-residue

small molecule ligands and solvents to be specified by hand.[34, 41] It also cannot be applied to other polymeric systems whose connectivity rules have not been pre-defined and coded up.[34]

A more general approach to identifying bond connectivities is presented by Zhang, et al. [42]. Atom pairs are identified as bonded if the distance between them meets the following criterion:

$$0.8 < d_{ij} < r_i + r_j + 0.4$$

where d_{ij} is the distance between atoms *i* and *j* and r_i , r_j are the covalent radii of atoms *i* and *j*. If more than 4 potential bonding partners for any non-hydrogen atom are identified using this approach, only the 4 bonds to the closest surrounding atoms are retained. However, this rule is not sufficient to distinguish between bonds of different orders, which requires additional rules to be applied.[39] Like all parameterized rules-based methods, this approach suffers from ambiguity in 'edge cases'. For example, long double bonds may be incorrectly classified as single bonds.[39] This particular method does not provide any information about molecular topology, only bond connectivity.[39]

The most general tool currently available for determining molecular topologies is embedded within the Automated Topology Builder (ATB) web server.[38] It takes as input molecular coordinates, bond connectivities and the formal charge on a molecule and returns both a non-redundant set of internal coordinates and associated force constant data.[38] It uses rules-based systems for defining rings and non-redundant sets of bond lengths, angles and dihedrals.[38] These rules are more general than fragment-based approaches like LEaP, but not universal; in ambiguous or ill-defined cases, manual intervention is required from the user.[38]

1.4. Force field parameters

Once molecular connectivities and topologies have been defined, corresponding force field parameters must be obtained.[27] For polymeric systems and solvent molecules, a lot of time and effort has been put into manually optimizing force field parameters for specific sets of structural subunits. For example, the manual for the commonly used Amber simulation package recommends ff14SB [43] for proteins, OL15 [44-45] for DNA and OL3[46-47] for RNA, GLYCAM for carbohydrates [48], and lipid14 for lipids [49]. For solvents, TIP3P[50] is the most commonly used water force field, although TIP-4P-Ew [51] and OPC [52]are being suggested as better optimised options. Each of these force fields contain a specialized pre-defined and hand-optimized set of parameters that feed into the general

potential energy functions described above to complete the definition of the potential energy surface for each system in question.

Within the ATB, force field parameters derived from quantum mechanical (QM) calculations can be obtained for general molecules, in a manner that is consistent with existing manually pre-parameterized force fields.[53-54] In particular, the ATB is designed to be interoperable with the GROMOS family of force fields[38]. However, ATB parameterization is typically limited to molecules with up to 40 atoms, due to the computational cost of the underlying QM calculations.[38] Further, this approach is limited to "bio-compatible" molecules, borrowing atom types and associated dihedral and non-bonded parameters from existing GROMOS force fields.[38] The ATB algorithm is completely incompatible with atoms that are not defined within GROMOS force fields.[38, 55-56]

1.5. Limitations of existing methods

The main problem with existing specialized force fields is that they are neither general nor generalizable.[48, 57] They cannot be applied universally, as there is no general purpose code for determining how to derive unique non-redundant internal coordinate sets for molecules of arbitrary topology, let alone a general and robust procedure for obtaining appropriate force field parameters.[57-58] They are also not generalizable, because each molecular system has its own unique molecular connectivity rule set and force field parameter set, and these must usually be hand-coded individually and specifically for each structural motif in a system.[41, 57] At best, the Automated Topology Builder software is a semi-automated user-guided process that makes the process of defining topologies and parameterizing force fields less manually intensive, but it is limited to relatively small molecules.[38]

A secondary disadvantage of existing force fields is the computational cost associated with evaluating the non-bonded interactions.[27] While this can be somewhat ameliorated by setting non-bonded interaction cutoffs, the sheer number of pair-wise non-bonded interactions makes this by far the most computationally intensive part of the overall energy and force evaluation procedure during a molecular dynamics simulation.[27] It is somewhat ironic that the terms that individually contribute least to the overall potential energy take by far the longest to evaluate.

1.6. New approaches

Access to richer topological information could be used to concurrently solve both of these problems. The molecular connectivity, topology and parameterization problem could be solved by automatically detecting repeating sub-units and expressing the energy of the sub-unit as a series expansion about a preferred geometry, or interpolating between a set of preferred geometries. For example, a topological force field for a protein might be defined as:

$$\mathrm{U(R)} = \sum_{\mathrm{bb}} \mathrm{U(R_{bb})} + \sum_{\mathrm{sc}} \mathrm{U(R_{sc})} + \sum_{\mathrm{bb,sc}} \mathrm{U(R_{bb}, R_{sc})} + \sum_{\mathrm{nb}} \mathrm{U(R_{nb1}, R_{nb2})}$$

where the first two terms capture short-range bonded interactions within backbone units (bb) and sidechain units (sc), respectively, the third term captures the effect of the side chain on the backbone conformation and energetics for connected sidechain-backbone units and the final term captures non-bonded and electrostatic interactions between all fragments that are not directly connected to one another (nb1, nb2).

However, this is just an example. A key feature of this approach would be the ability to define any energetically characterise any repeating structure motif, or indeed, any unique structural motif whether it repeats or not.

Clearly, there would be far fewer terms in such a topological force field than pair-wise atomic interactions in a conventional force field, so the number of terms to evaluate would be less and so take less time, although the complexity of calculating each individual term may increase somewhat. The fragmented nature of this force field ansatz would also make it better suited to robust automated parameterization from ab initio quantum chemical data.

1.7. Aim of the present work

However, before this can even be attempted, the first step, and the aim of this work, is to develop a simple, general and robust automated algorithm for determining connectivities and topologies of a wide variety of molecular systems.

2. Methods

2.1 Overview

Automatically detecting molecular connectivities and topologies, without requiring user input or defining system-specific sets of rules, is a challenging algorithmic problem.[39] The novel strategy developed in this work relies on two overlapping, or near equivalent, assumptions; one implicit and one explicit. The explicit assumption is that the closer two atoms are the more likely they are to be involved in a bonding interaction. The second implicit assumption is that the atomic positions are chemically sensible and exist in a stable "chemically reasonable" configuration where bonds are not actually in flux.

This work also relies on the novel insight that it should be simpler to create more bonds than necessary and remove those that are not physically meaningful, rather than to pre-define a rigid set of rules that only allows physically meaningful bonds to be created in the first place. Similarly, it should be easier to find rings by removing non-ring components than pre-defining rules to identify them.

In brief, there are four key stages to determining molecular connectivities and topologies:
1. Read in and store atomic coordinates and identities
2. Process atomic coordinate data to find all physically meaningful bonds
3. Process bond connectivity information to identify connected ring systems
4. Assign and store formal bond types and charges.

The data structure used to store and reference the data generated during this process is illustrated in Figure 2.1, and detailed explanations of each step in the new algorithm are explained below.

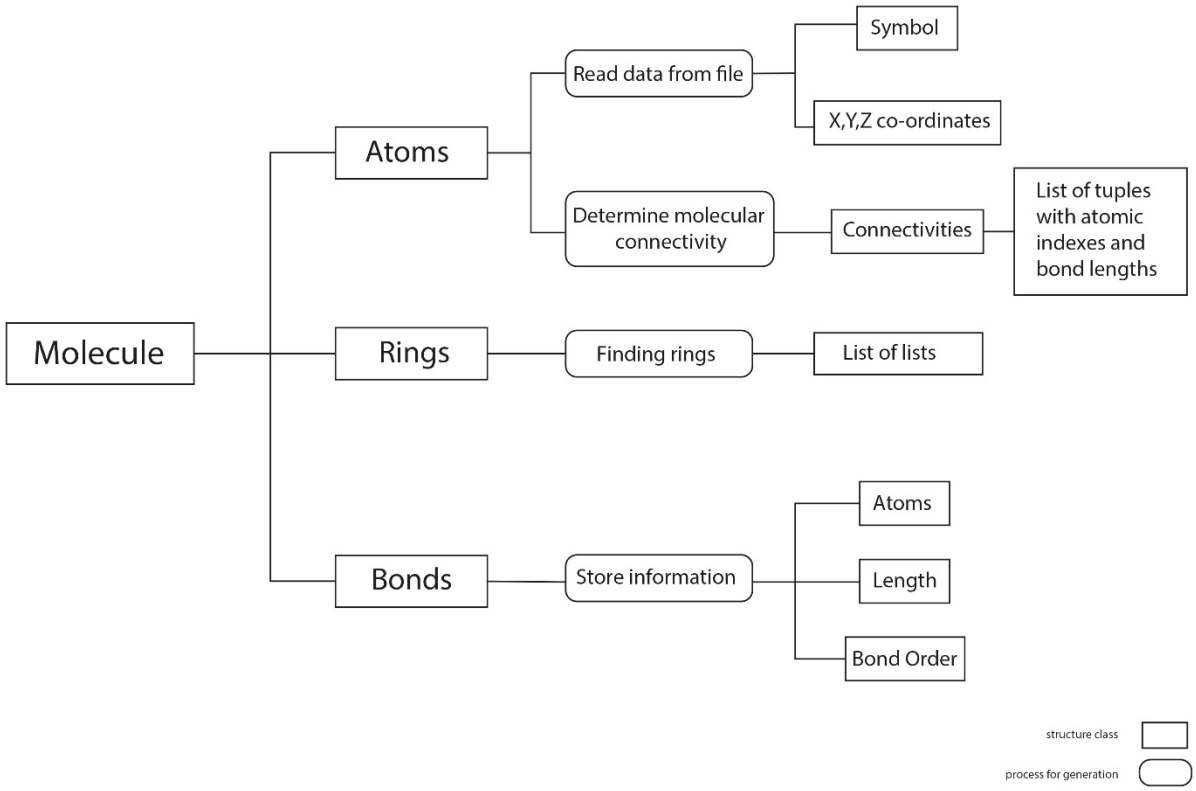


Figure 2.1: The structure and hierarchy of Python classes used for structural elucidation. Each molecule contains atoms, rings and bonds. Each atom is defined by its atomic symbol and Cartesian coordinates, and connections to other atoms which are automatically generated and stored as connected atom index and connection distance. Rings are defined as collections of connected atoms. Bond data is stored as atom pairs along with associated bond lengths and orders.

2.2. Parsing data from files

The two major file types used in this project are XYZ and PDB. XYZ files contain the bare minimum data required for the algorithm to function. PDB files are designed for capturing more meta-data related to protein structures, and so are a more prolific data storage format. However, only the atomic symbols and Cartesian (x,y,z) coordinates are initially read and used to instantiate the molecule class and populate the atom data structure.

2.3. Determining molecular connectivities

The molecular connectivity algorithm illustrated in Figure 2.2 is the cornerstone of this work. It is designed to identify all chemically-connected bonds within a system, without relying on any external parameters or rule sets.

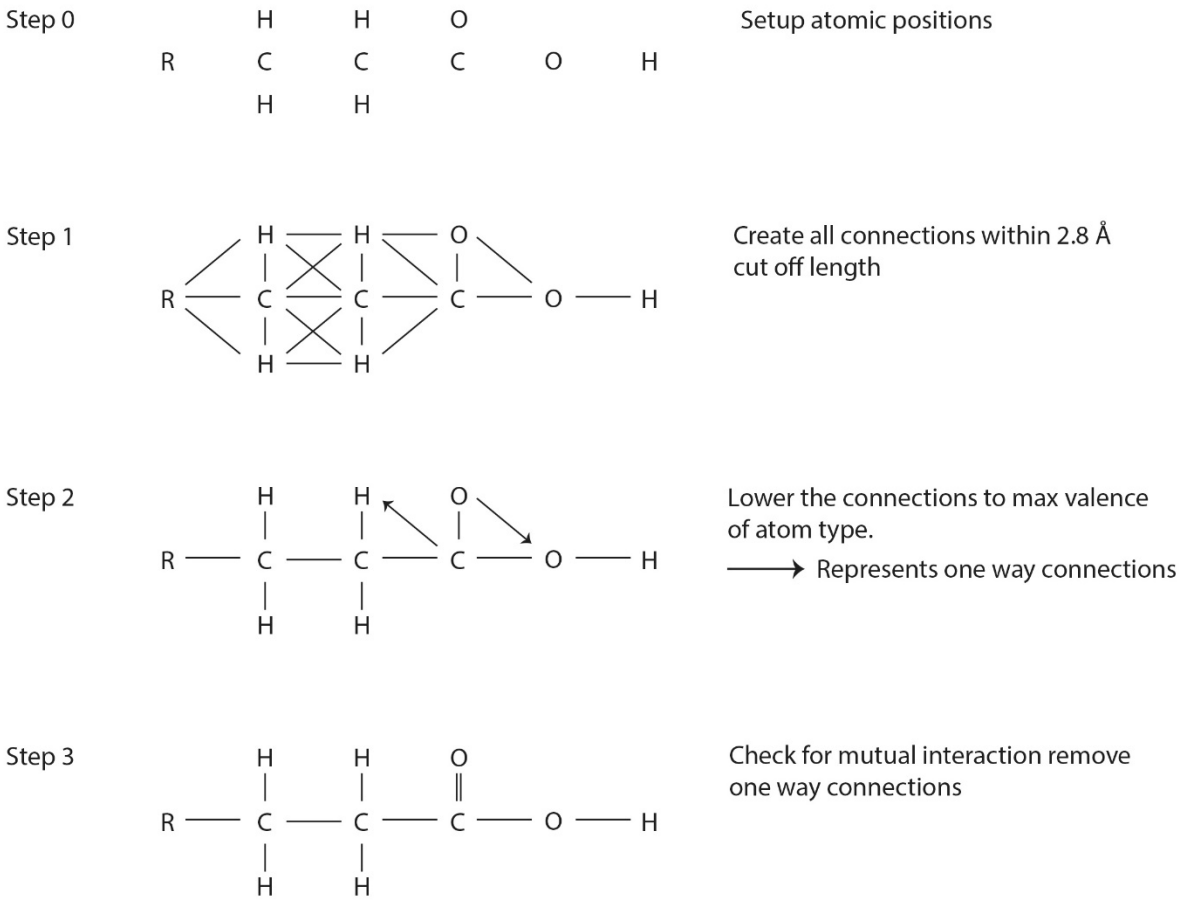


Figure 2.2: Substituted propanoic acid is used as an illustrative example of the new molecular connectivity algorithm. In the first step, all possible connections to all atoms within a 2.8 Angstrom radius are formed. In the next step, only the closest "chemically sensible" set of bonds are retained. Finally, all one-way connections are removed during a mutual connectivity checking process. Bond orders will be increased if two connected atoms both have empty valences at the conclusion of the entire process.

A more detailed outline of the algorithm is provided as pseudocode below.

```
------------------------------------------------------------
Chemical bond detection algorithm
------------------------------------------------------------
# Data
expected_valence = dictionary of chemical valencies by atom type
bond_length_cutoff = 2.8 if first_pass else 2.0

# Assign maximum chemical valences to each atom, by type
for atom in molecule.Atoms:
  atom.ExpectedValence = expected_valence(atom.Symbol)

# Find superset of possible bonds for each atom
for i,atom1 in enumerate(molecule.Atoms):
  for j,atom2 in enumerate(molecule.Atoms):
     if j > i:
       if distance(atom1,atom2) < bond_length_cutoff:
         append [distance,atom2] to atom1.Connectivity list
         append [distance,atom1] to atom2.Connectivity list

# Sort connectivity lists in order of increasing bond length, and truncate
to "chemically sensible" possibilities
for atom in molecule.Atoms:
  atom.Connectivity = atom.Connectivity.sort()[:atom.ExpectedValence]

# Check for and remove one-way connections
for atom1 in molecule.Atoms:
  for [distance,atom2] in atom1.Connectivity:
    paired = False
    for atom3 in atom2.Connectivity:
      if atom1 == atom3
        paired = True
    if not paired:
      remove atom2 from atom1 connectivity list
      reduce valence of atom1 by 1
------------------------------------------------------------
```

2.4. Ring finding

The novel ring-finding algorithm developed in this work is based upon iteratively removing non-ring atoms and previously identified rings. The key quantity used to identify non-ring atoms is the "heavy valence connectivity number" illustrated in Figure 2.3. Hydrogen atoms and bonds to them are ignored, as hydrogen atoms cannot form rings in conventional bonding situations.

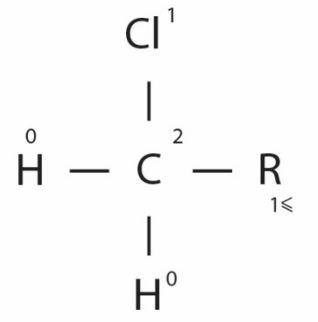


Figure 2.3. Example heavy valence connectivity numbers for substituted chloromethane. The heavy valence connectivity number is defined the number of "heavy" (non-hydrogen) atoms connected to each heavy atom itself. For example, the central carbon atom has a heavy valence connectivity of 2 as it is bound to both chlorine and the non-hydrogen R group. Chlorine has a heavy valence connectivity of 1, as it is only bound to the carbon atom. The R group has a heavy valence connectivity of at least 1, which could increase depending on the nature of the R group.

The ring-finding algorithm iterates over three key steps:

(a) Remove chain ends

(b) Define connected groups of atoms each with identical heavy valence connectivity numbers: "proto-rings" (heavy valence connectivity = 2) and "capping atoms/groups" (heavy valence connectivity > 2)

(c) Identify capping atoms for each proto-ring, store identity of atoms in ring, remove terminal ring atoms

These steps are repeated until no atoms are left or there is no change between iterations. If there is no change between iterations, the connectivity algorithm is re-run with a shorter bond length cutoff remove spurious intra-ring bonds that may have formed and the ring-finding process repeated. If there is no change between iterations a second time, the algorithm terminates and returns the identities of the atoms whose topology could not be assigned.

The algorithm's three steps in the ring-finding process will be described in detail below.

(a) Removal of non-ring chains

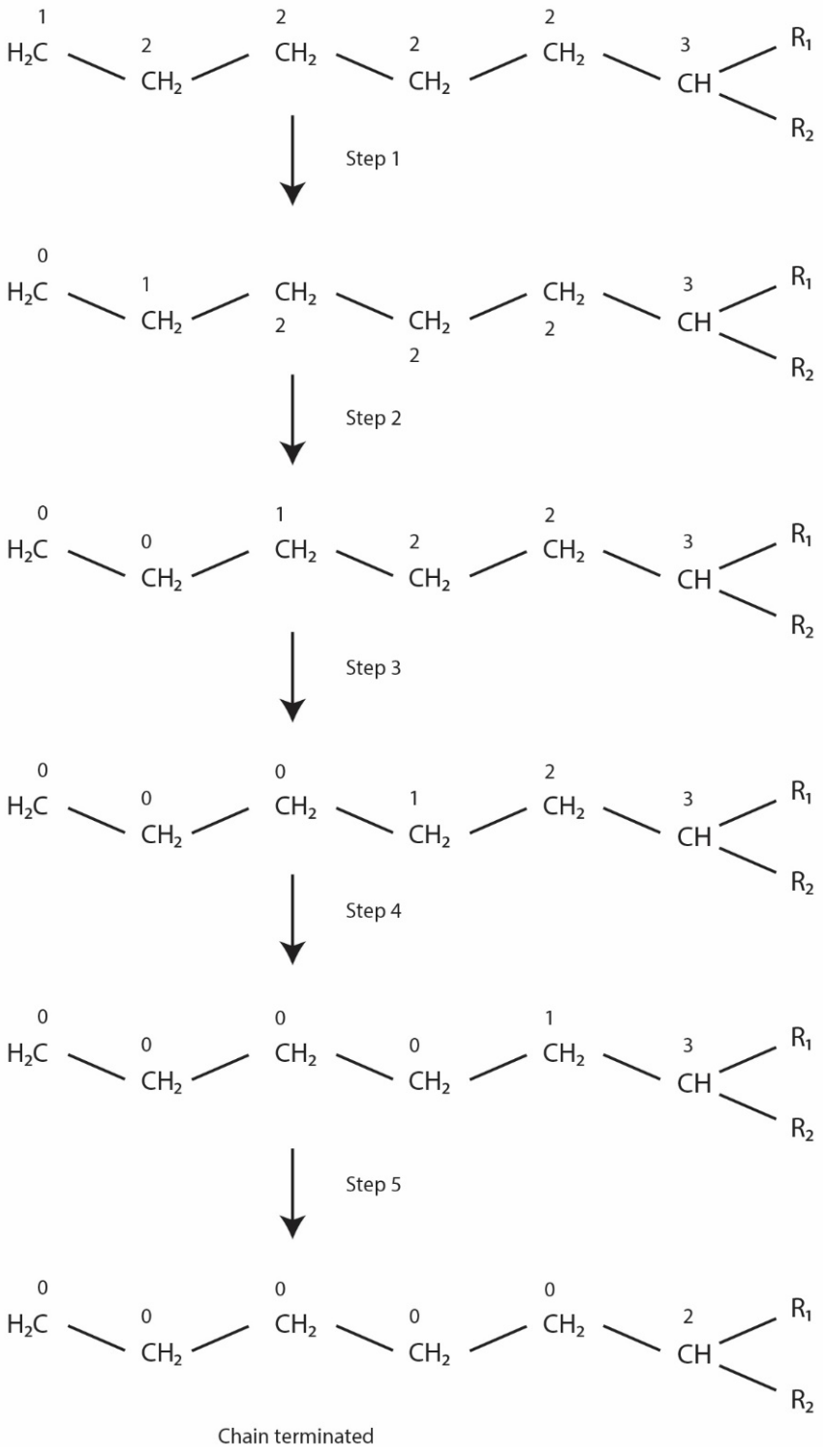


Figure 2.4: Illustration of the process by which chains are iteratively reduced, until a junction is arrived at.

Chain ends in the molecule are identified as atoms with heavy valence connectivity numbers of 1. The heavy valence connectivity count of these atoms, along with their bonded neighbours is reduced by 1, a process that iterates until there are no further chain ends to be found. This process is illustrated in Figure 2.4, and detailed in the following pseudocode. A chain is considered removed when the heavy valence connectivity numbers of all of its constituent atoms have been processed down to 0.

```
------------------------------------------------------------
Chain removal algorithm
------------------------------------------------------------
# Compute heavy valence connectivity numbers
for atom1 in molecule.Atoms:
  atom1.HeavyValence = 0
  for [distance,atom2] in atom1.Connectivity:
    if atom2.Symbol not equal to 'H':
      atom1.HeavyValence += 1

# Identify chain ends
ChainEnds = []
for atom in molecule.Atoms:
  if atom.HeavyValence = 1:
    ChainEnds.append(atom)

# Iterative procedure to reduce chain
while len(ChainEnds) is not 0:
  NewChainEnds = []
  for atom in ChainEnds:
    atom.HeavyValence -= 1
    for [distance,atom2] in atom.Connectivity:
      if atom2 not 'H':
        atom2.HeavyValence -= 1
        if atom2.HeavyValence == 1:
          NewChainEnds.append(atom2)
  ChainEnds = NewChainEnds
------------------------------------------------------------
```

(b) Group formation

The group formation algorithm essentially just finds sets of connected atoms with the same heavy valence connectivity number. While this is easy to say, it is much harder to achieve in a general, robust and automated manner. A single example iteration of the group formation process in illustrated in Figure 2.5. All groups formed across a collection of topologically diverse ring systems are illustated in Figure 2.6, and the overall algorithm is outlined in pseudo-code form below that.

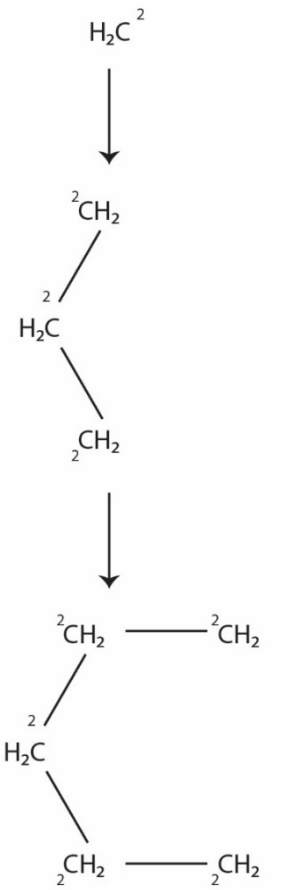


Figure 2.5: Visualisation of how groups are created. This process is repeated for all possible choices of initial ring atom.

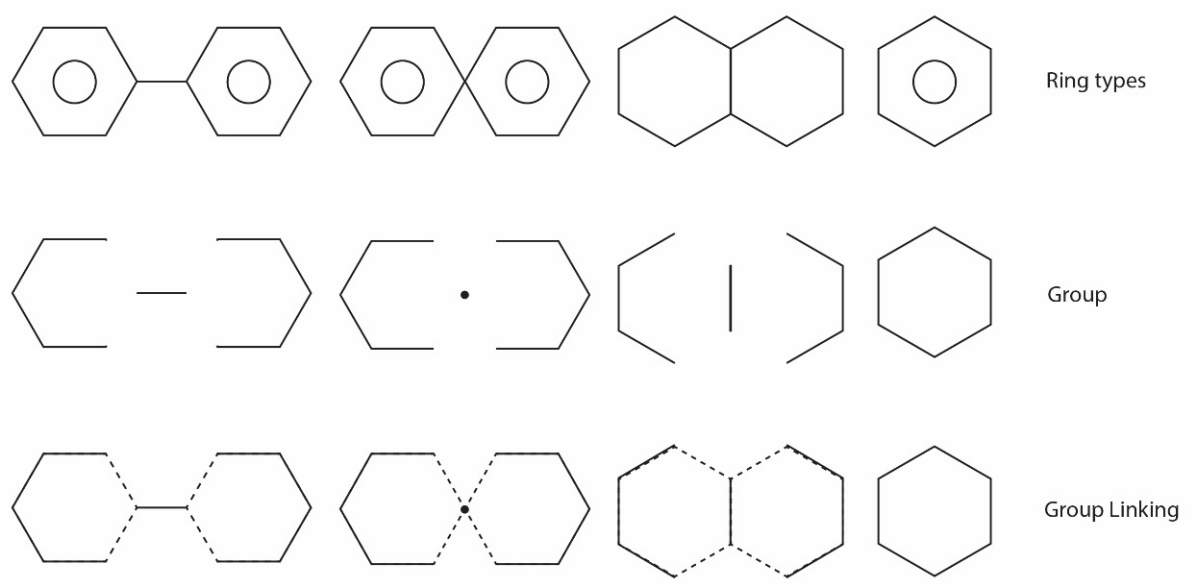


Figure 2.6: Result of group formation process for benzene, biphenyl, spiro[5.5]undecane, and naphthalene.

```
------------------------------------------------------------
Group formation algorithm
------------------------------------------------------------
# Make group_set object to store all identified groups, start each group
with seed atom, keep adding connected atoms until no more of the same heavy
valence are to be found

group_set = set()

for seed_atom in molecule.Atoms:
  if seed_atom.HeavyValence > 0:

    atoms = current_atoms = [seed_atom]

    n = 0
    while len(atoms) > n:
      n = len(atoms)
      for current_atom in current_atoms:
        for [distance,connected_atom] in current_atom.Connectivity:
          if atom.HeavyValence == connected_atom.HeavyValence:
            atoms.add(connected_atom)
```

```
      current_atoms = updated copy of atoms list

    atoms = final sorted copy of atoms list

    group_set.add(group(atoms, atom.HeavyValence))

Remove redundant copies of groups (same group from different seed atoms)

group_list = list of unique groups from group_set

------------------------------------------------------------
```

At the end of the group formation process, group_list contains only unique groups, each identified by the atoms within them and the heavy valence number of those atoms.

(c) Identification of rings

Ring identification proceeds in two steps:

1. Identify externally-connected atoms and/or groups for each proto-ring (groups with heavy valence connectivity = 2). If no external connections can be found, an isolated ring has been created directly during group formation. It is immediately stored as a new ring, and removed by setting the heavy valence connectivity numbers of all constituent atoms to 0.

2. Process heavy valence connectivity numbers of atoms in proto-ring and capping group, and store identity of rings removed in this process.

This procedure is illustrated in Figure 2.7 for an isolated ring system tethered to the rest of the molecule by a chain. Figure 2.8 deals with the case of a fused ring system. The outcome of the overall ring processing procedure for a series of topologically diverse ring systems is shown in Figure 2.9.

The subsequent pseudocode outlines the logical processes involved in ring formation and data storage. Both the group formation and ring identification processes are repeated iteratively until the heavy valence numbers of all atoms have been reduced to zero (all atoms assigned to either chains or rings), or the algorithm fails to converge. Convergence failure typically indicates that complex fused ring systems are present.

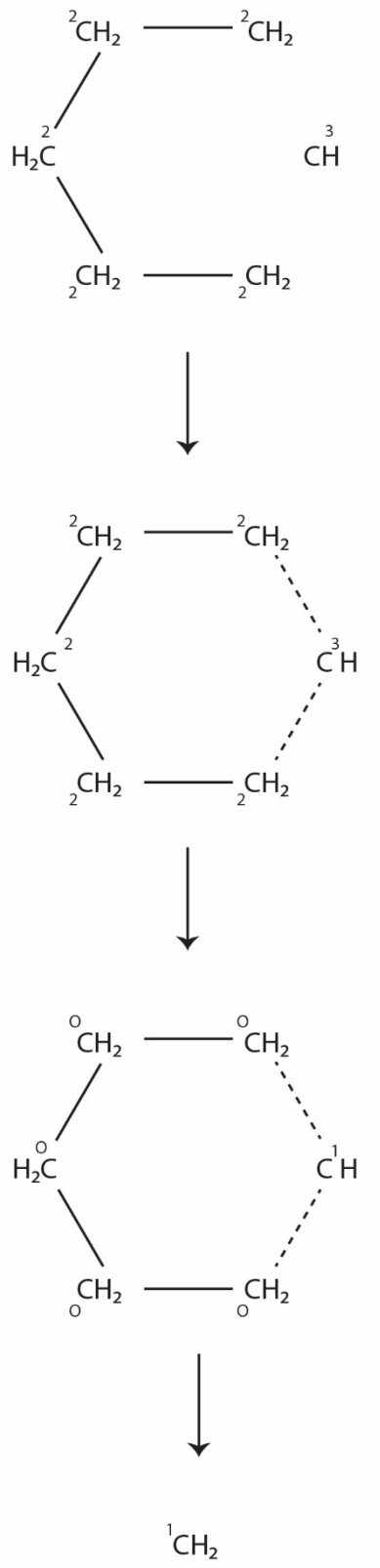


Figure 2.7: Ring processing procedure for a terminal ring connected to the rest of the molecule by a chain.

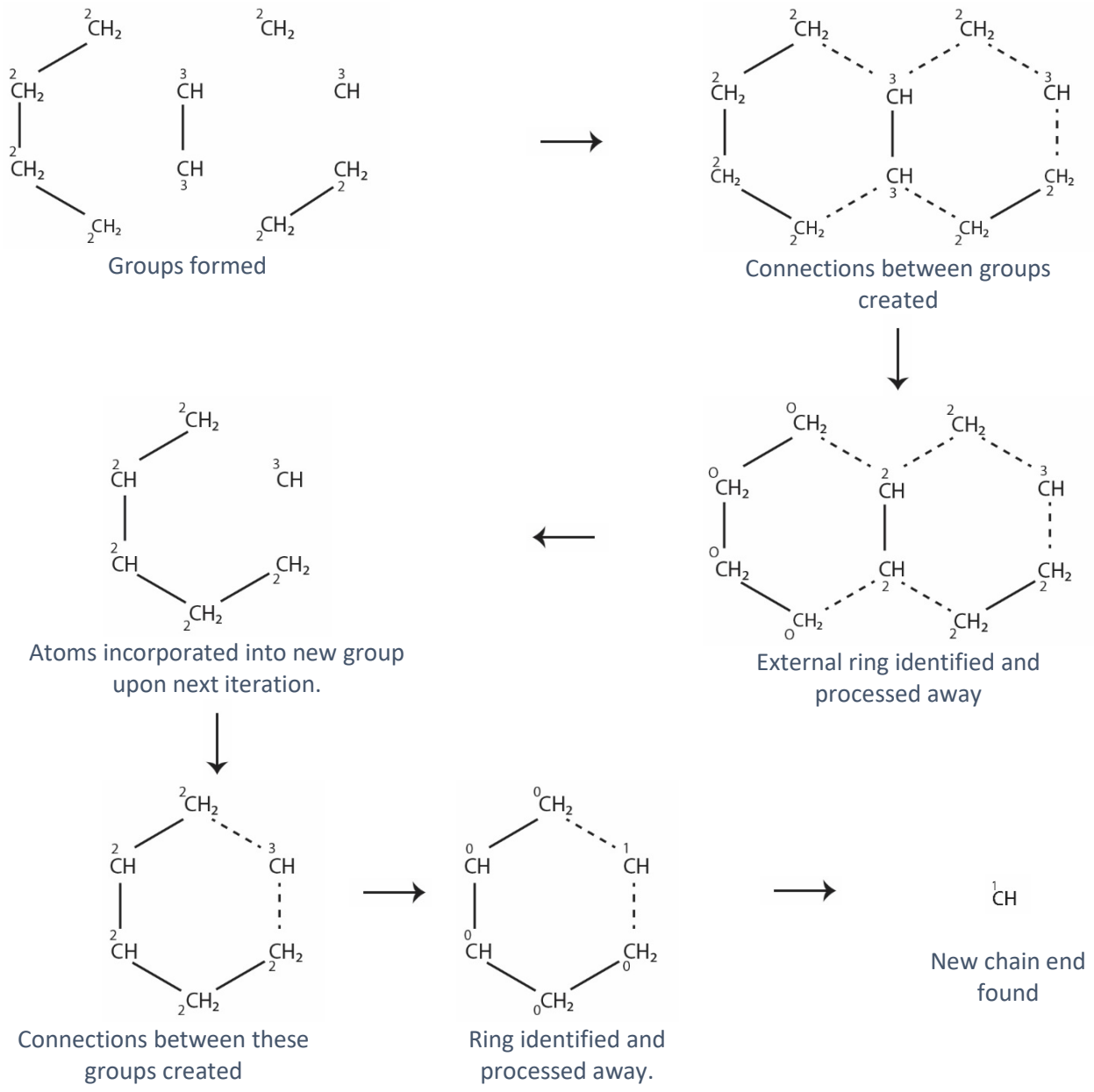


Figure 2.8: Ring processing procedure for a terminal ring embedded in a fused ring system, which is connected to the rest of the system through the mono-hydrogenated carbon atom with heavy valence = 3.

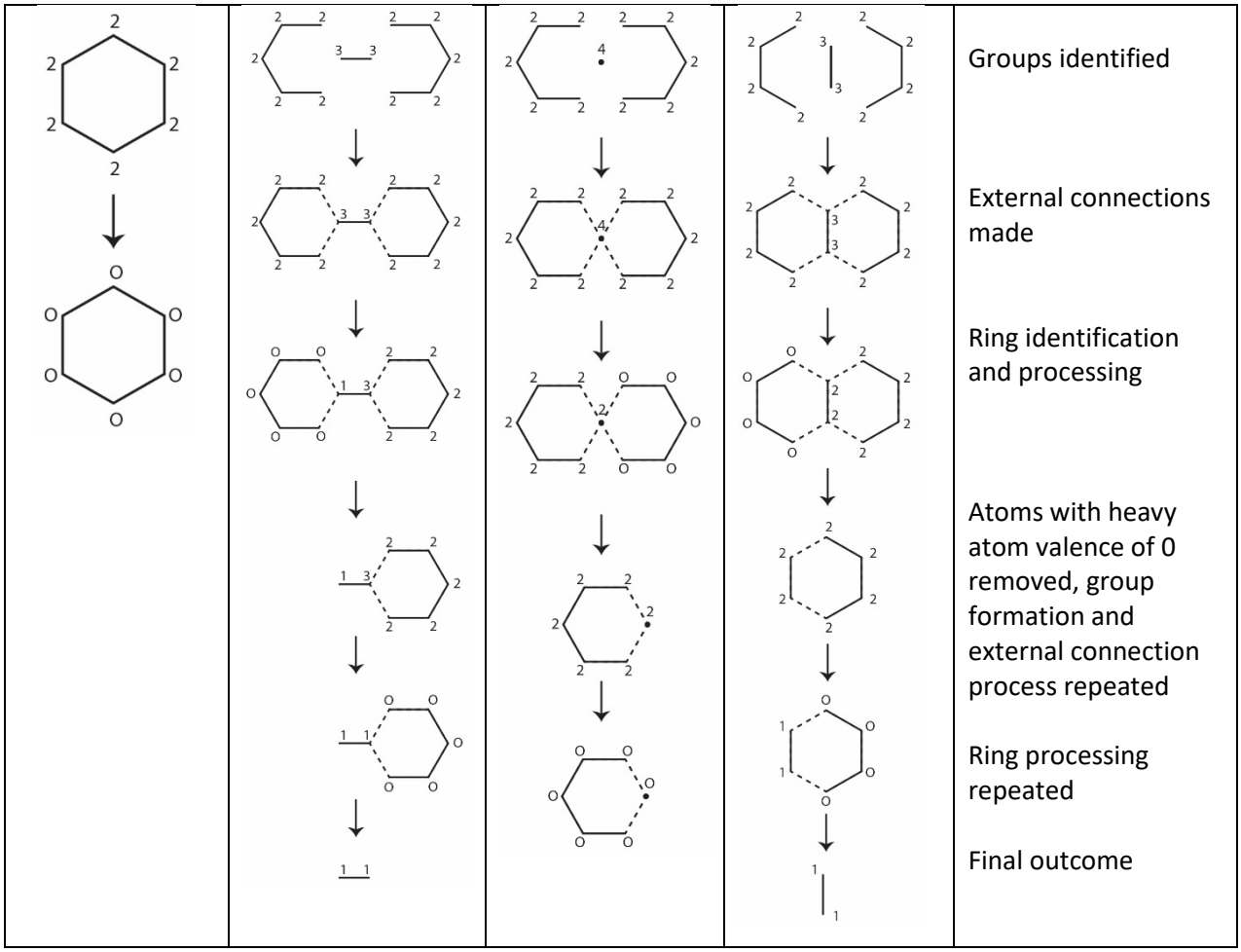


Figure 2.9: Intermediate and final outcomes of ring processing procedure for benzene, biphenyl, spiro[5.5]undecane, and naphthalene.

```
------------------------------------------------------------
Ring identification and data storage
------------------------------------------------------------

# Divide groups up into proto-rings and capping groups

proto_rings = [group for group in group_list if group.HeavyValence == 2]
capping_groups = [group for group in group_list if group.HeavyValence > 2]

# Find and store identities of externally-connected atoms for each proto-
ring, and then identify and store which capping group they are members of

for group in proto_rings:
  atomic_connections = set()
  group_connections = set()
  for atom in group.Atoms:
    for [distance,conn_atom] in atom.Connectivity:
      if conn_atom.HeavyValence > 0 and conn_atom not in group.Atoms:
        atomic_connections.add(conn_atom)
        for conn_group in capping_groups:
          if conn_atom in conn_group:
            group_connections.add(conn_group)
  group.ExternalAtoms = atomic_connections
  group.ExternalGroup = group_connections

# Process and store ring systems on a case by case basis

for group in proto_rings:
  ring = []
  if len(group.ExternalAtoms) == 0:
    ring = group.Atoms
  elif len(group.ExternalAtoms) == 1:
    ring = group.Atoms + group.ExternalAtoms
  elif len(group.ExternalGroup) == 1:
    for atom in group.ExternalAtoms[0].Connectivity:
      if atom = group.ExternalAtoms[1]
        ring = group.Atoms + group.ExternalAtoms
      else:
        ring = group.Atoms + group.ExternalGroup[0]
  else:
    print('Inconclusive: no ring identified, evaluate next group’)
------------------------------------------------------------
```

2.5. Bond redefinition and characterisation

Once the connectivity and ring finding algorithms are finished, the bonds are redefined and abstracted into a separate bond class. This allows easier access to bonding data, and facilitates characterization and visualization of the identified bonds within the macromolecule, complex or molecular assembly. This process is illustrated in Figure 2.10 for a simple ring system, a simple ring and chain, and a more complicated case in which bond order assignments are initially ambiguous, requiring a two-step process to disambiguate.

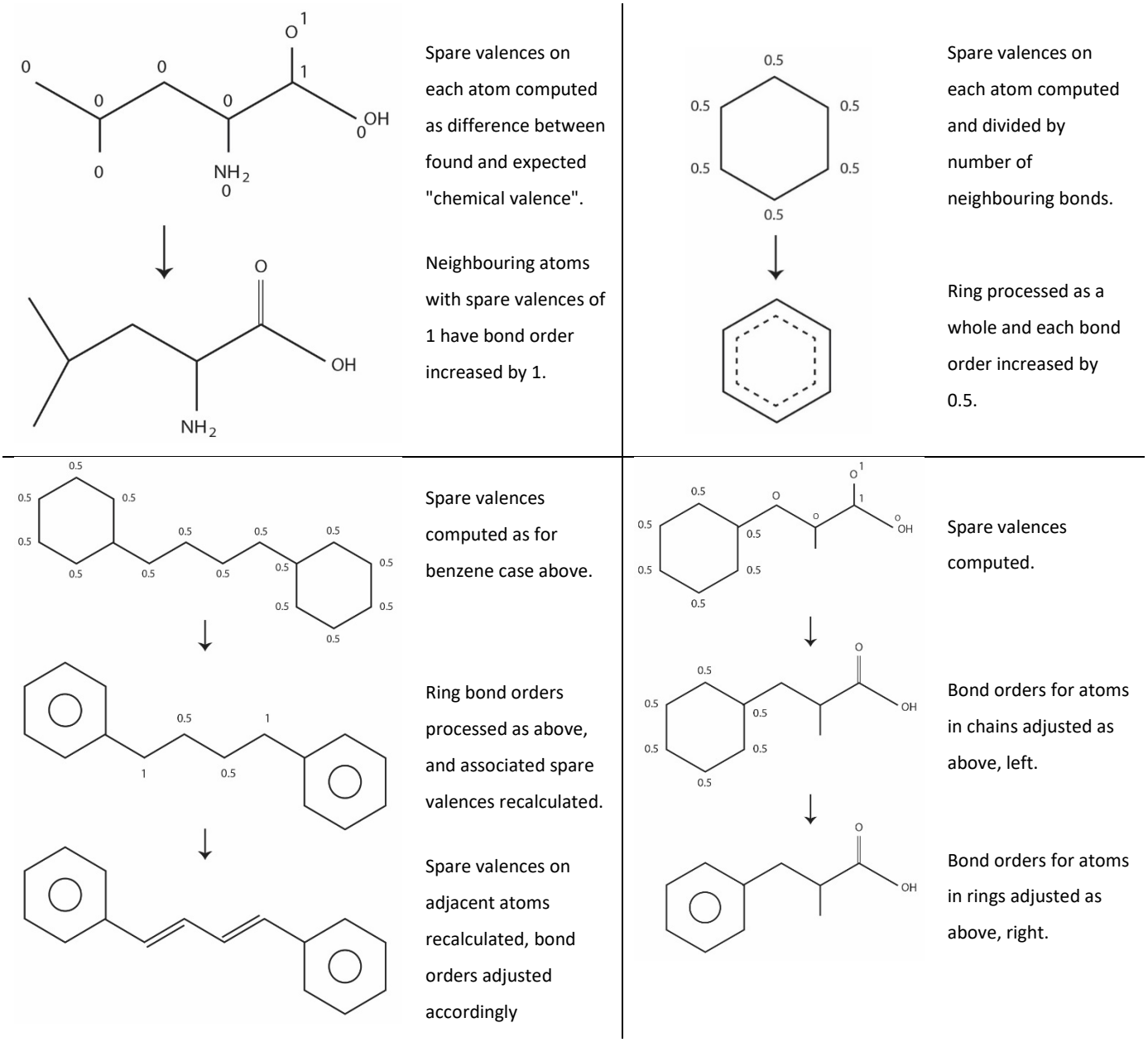


Figure 2.10: Illustration of bond order assignment process for leucine, benzene, a phenyl-terminated conjugated chain, and 2-benzylpropanoic acid.

With bond order and atomic valence data in hand, formal charges are computed and assigned by comparing the observed atomic valences with expected. Examples are given in Figure 2.11.

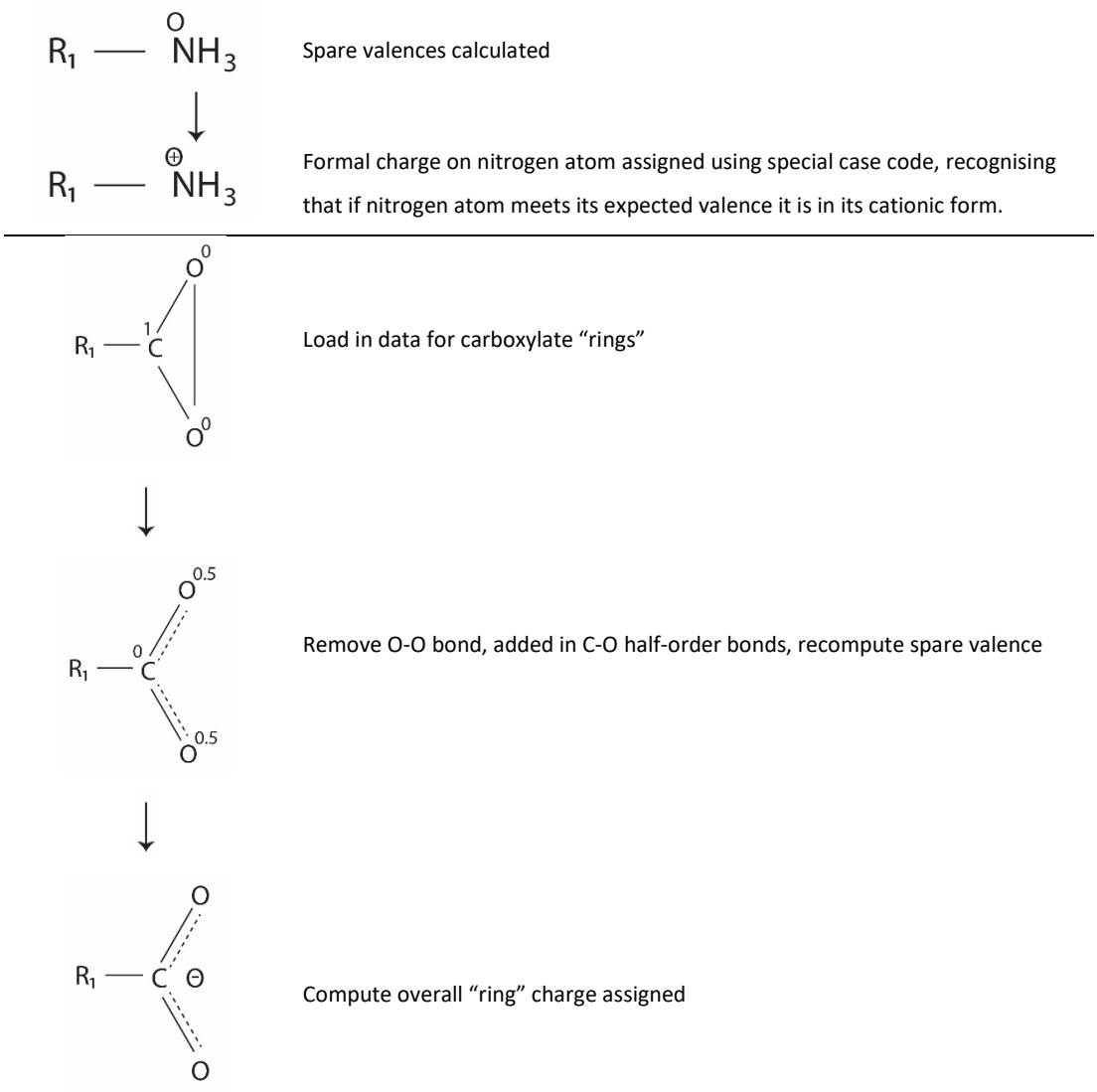


Figure 2.11: Examples of charge assignment for quaternary ammonium and carboxylate-containing species. Nitrogen atoms are treated differently to other atoms on the periodic table, and assigned an expected valence that is not the same as the chemical valence of the uncharged atom. For nitrogen, the expected valence is set to 4, corresponding to the cationic species, i.e. the very common protonated or quaternary amine case. Hence, when the found valence of a nitrogen atom matches its expected valence, the atom is assigned a formal positive charge. For all other systems, charges are assigned to satisfy unfilled spare valences, as illustrated for the carboxylate anion above.

Pseudocode describing the overall bond order and formal charge assignment process is provided below.

```
------------------------------------------------------------
Bond data processing, bond order and partial charge assignment
------------------------------------------------------------

# Reformat atom connectivity data into bond class data format
# Note that a "heavy bond" is a bond between 2 "heavy" (non-H) atoms
for atom in molecule.Atoms:
  bonds = []
  heavy_bonds = []
  for [distance,conn_atom] in atom.Connectivity:
    bond = Bond(Distance=distance,Atoms=[atom,conn_atom],Order=1)
    bonds.append(bond)
    if atom != 'H' and conn_atom != 'H': heavy_bonds.append(bond)
  atom.Bonds = bonds
  atom.HeavyBonds = heavy_bonds

# Compute spare valences on each atom
for atom in molecule.Atoms:
  atom.FoundValence = len(atom.Connectivity)
  atom.SpareValence = atom.ExpectedValence - atom.FoundValence

# Compute number of potential partners for multiple bond formation
for atom in molecule.Atoms:
  for conn_atom in atom.Connectivity:
    if conn_atom.SpareValence > 0.5:
      atom.Options += 1

# Compile list of all bonds and "heavy bonds" in molecule for convenience
molecule.Bonds = []
molecule.HeavyBonds = []
for atom in molecule.Atoms:
  molecule.Bonds.extend(atom.Bonds)
  molecule.HeavyBonds.extend(atom.HeavyBonds)

# Increment bond order for isolated unsaturated bonds
for bond in molecule.HeavyBonds:
  if bond.Atoms[0].Options == bond.Atoms[1].Options == 1:
    if bond.Atoms[0].SpareValence == bonds.Atoms[1].SpareValence:
```

```
      bond.Order += bond.Atoms[0].SpareValence
      for atom in bond.Atoms:
        atom.SpareValence = 0
        atom.FoundValence += bond.Atoms[0].SpareValence
      else:
        MinSpareValence = min([atom.SpareValence for atom in bond.Atoms])
        bond.Order += SpareValence
        for atom in bond.Atoms:
          atom.SpareValence = -MinSpareValence
          atom.FoundValence += MinSpareValence

# Adjust bond orders for conjugated systems, starting with rings
for atom in molecule.Atoms:
  atom.SpareValence = atom.SpareValence/atom.Options if atom.Options != 0

for ring in molecule.Rings:
  for atom in ring:
    for bond in atom.HeavyBonds:
      bond.Order += atom.SpareValence
      for bound_atom in bond.Atoms:
        bound_atom.SpareValence = 0
        bound_atom.FoundValence += atom.SpareValence

# Then moving on to alternating unsaturated chains
WholeSpareValences = Count(atoms in molecule with SpareValence = 1)
while WholeSpareValences > 0:
  for bond in molecule.HeavyBonds:
    for i,atom in enumerate(bond.Atoms):
      if atom.SpareValence = 1:
        bond.Order += 1
        other_atom = bond.Atoms[i-1]
        other_atom.FoundValence = len(other_atom.Connectivity) + 1
        other_atom.SpareValence = other_atom.ExpectedValence -
                                  other_atom.FoundValence
        for bond in other_atom.HeavyBonds:
           bond.Atom[1].SpareValence = 1
  WholeSpareValences = Count(atoms in molecule with SpareValence = 1)

# Special-case code for dealing with bond orders in carboxylate "rings"
for ring in molecule.Rings:
  if len(ring) == 3:
```

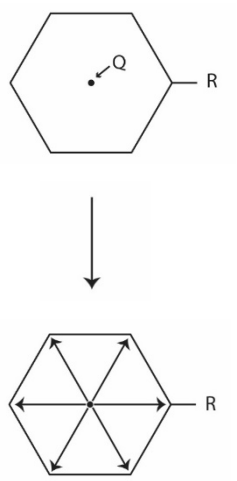
```
    atom_symbol = [atom.symbol for atom in ring]
    if atom_symbol.sort() == ['C','O','O']
      for atom in ring:
        for bond in atom.Bonds:
          if bond.Atoms[0] == bond.Atoms[1] == 'O':
            bond.Order = 0
          else:
            bond.Order = 1.5

# Derive formal charges for each atom
for atom in molecule.Atoms:
  atom.SpareValence = atom.ExpectedValence - atom.Bond.Order
  atom.Charge = -atom.SpareValence
  if atom.symbol == 'N': atom.Charge += 1

# Assign formal charges for each ring
for ring in molecule.Rings:
  ring.Charge = sum([atom.Charge for atom in ring])
```

2.6. Computing and storing derived ring data

Once the rings have been found using the above ring identification techniques, additional ring properties are computed and stored for convenience. In particular, a dummy atom along with dummy bonds are associated with each ring centroid, as illustrated in Figure 2.12. This is a simple geometric process that does not require any further explanation or elucidation.



Average the Cartesian coordinates of all atoms in the ring said this is the position of a dummy Q atom.

Set Q atom expected valence to be equal to the number of ring atoms, then form one-way bonds between this atom and all ring atoms.

Figure 2.12. Illustration of derived ring data: a dummy atom, Q, is placed at the "heavy atom" centre of mass, and one-way bonding connections from this atom out to all constituent ring atoms are defined.

2.7. Source code available

Source code for a full implementation of the algorithms outlined above is available at:

https://github.com/craig260/connectivity-ringfinding-algorithm

3. Validation and testing: Biomolecules

3.1. Introduction

The generality, effectiveness and correctness of the algorithm described in Chapter 2 can only be confirmed through a rigorous and extensive external validation process. A number of strategies are possible:

1. Manual inspection of all results,
2. Comparison with results obtained from other, rule-based codes, or
3. Verification against additional topological data available from a trusted external source.

The first strategy is clearly impractical for large scale testing, and inconsistent with the automated, unsupervised nature of the overall process. The second strategy risks importing the biases encoded within rule based codes and is also surprisingly difficult to implement, because other codes do not generally output connectivity and topological information in an easily accessible way.

Biomolecular structures accessible through the Research Collaboratory for Structural Bioinformatics (RSCB) protein data bank (PDB)[59] are an ideal external validation data source, as PDB files contain additional meta-data on the types of fragments present in a system (amino acids, nucleic acids), from which connectivity and topological information can be inferred.

However, the vast majority of structures in the PDB are obtained via X-ray crystallography[60-61], which is unable to detect the positions of hydrogen atoms within the structure, which are therefore generally omitted.[62] The algorithm presented in Chapter 2 relies on all atoms within a structure being present and correct in order to establish connectivity patterns that fulfil chemical valences in a internally-consistent manner.

Fortunately, three-dimensional structures for small proteins and peptides in solution can also be obtained via nuclear magnetic resonance (NMR) spectroscopy, by matching simulated and observed NMR spectra during structural refinement.[63-64] The main advantages of this technique are the ability to study proteins in their native state and obtain full sets of molecular coordinates for all atoms in the biomolecule, including hydrogens[65]. A set of 1,502 such structures was extracted from the PDB

and will be used as the testing and validation set for the remainder of this chapter. These systems range in size from 64 – 13582 atoms, and contain one or more of the following components: amino acids, nucleic acids, and synthetic linkers. Their PDB accession codes are available in Appendix A.

3.2. Methods

3.2.1. Analysis and validation procedure

The overall procedure for checking and characterizing outcomes is illustrated in Figure 3.1.

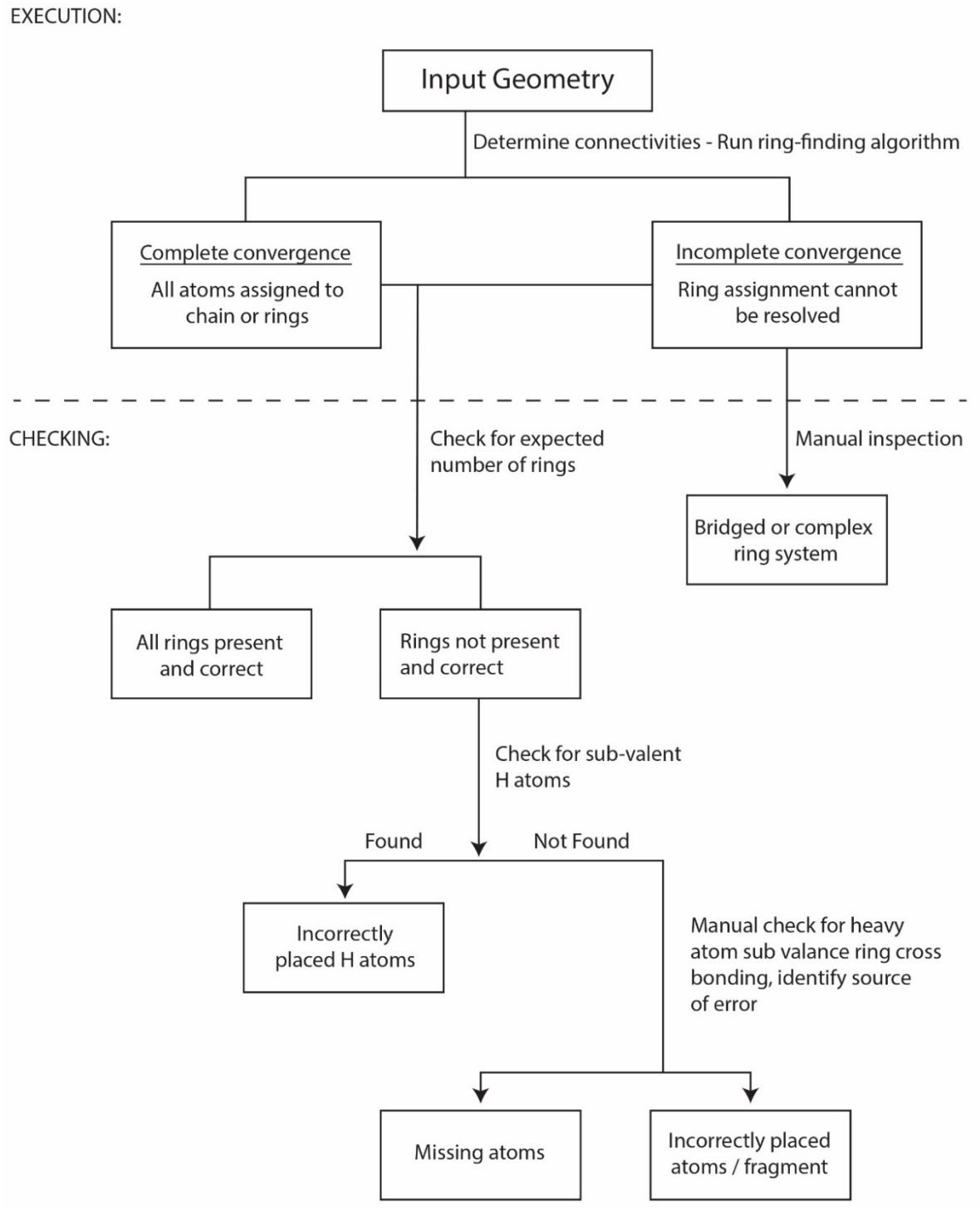


Figure 3.1: Flow chart illustrating outcomes (in boxes) of algorithm execution and checking processes (connecting lines and their labels). Branch points lead to mutually exclusive outcomes.

After the user inputs the initial geometry, the algorithm will proceed until there is no change in assigned atomic valence numbers from one step of the ring-finding code to the next.

If the algorithm converges completely, all atoms within the molecule end up assigned to either rings or chains. If the algorithm converges incompletely, some atoms remain unassigned, but no further progress is possible. In this case, the remaining atoms are printed to file for the user to inspect. They are expected to be part of topologically complex ring systems, e.g. bridged rings, or rings within macrocycles. No further automated checking of these systems is carried out.

For all molecules that completely converge, an automated checking procedure is performed, in which the number of expected rings (from PDB meta-data) is compared with the number of found rings. If these match, no further analysis is required. However, if the number of rings found is less than expected, more detailed analysis is carried out to identify the underlying cause, stemming from incorrect input data.

3.2.2. Algorithmic modification: disulphide bridges

During initial testing, it was found that a larger proportion of molecules than expected failed to completely converge – approximately 100/1500. Upon visual inspection, it was found that most of these involved macrocycles formed through cysteine cross-linking, as illustrated in Figure 3.2.

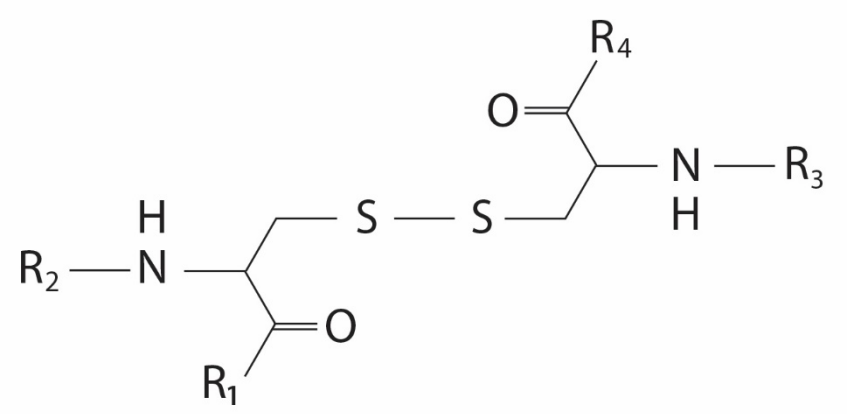


Figure 3.2: Disulphide (S-S) bond formation between cysteine residues results in the formation of intramolecular macrocycles when both cysteines form part of the same protein backbone (R_1 connects with R_3 or R_2 connects with R_4).

If the protein backbone section joined by the cysteine cross-link also contained proline rings, the resultant ring-within-a-macrocycle topology could be disambiguated, so incomplete convergence occurred.

Computationally, this situation is avoided by initially “breaking” the disulphide bond, running the ring-finding code, and then “recreating” the disulphide bond at the end of the process.

3.3. Results & Discussion

3.3.1. Case study – Cyclic tetrapeptide ALA-ARG-ALA-linker

The cyclic tetrapeptide illustrated in Figure 3.3 was chosen as a suitable model system to illustrate how the algorithm works because it is relatively small and easy to visualise yet still exhibits a substantial degree of topological complexity.

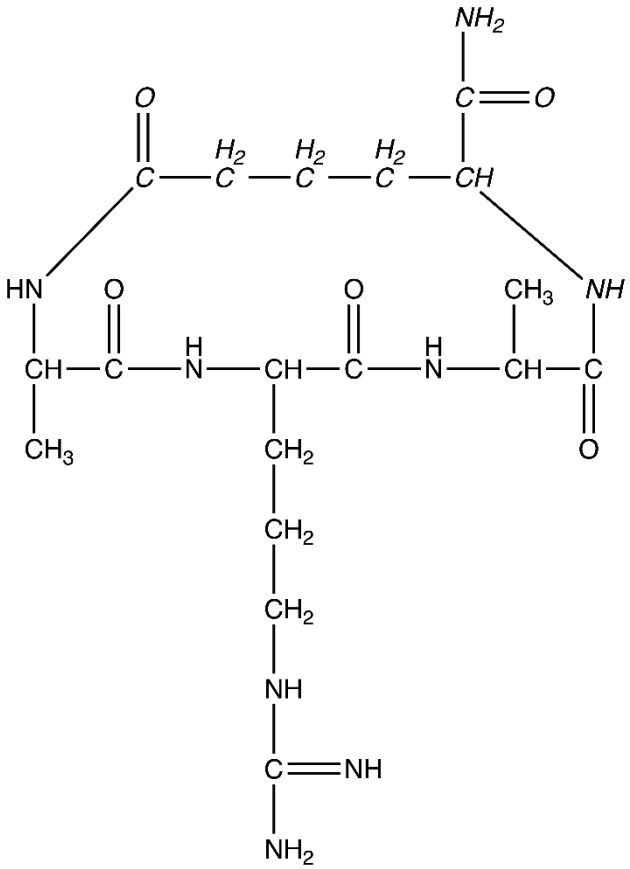


Figure 3.3: Canonical chemical line drawing of cyclic tetrapeptide ALA-ARG-ALA-linker. The atoms that comprise the non-natural linker are indicated in *italic* font.

The connectivity detection process is illustrated in Figure 3.4. At the end of this process, the assigned bonding patterns are chemically reasonable, although not always chemically conventional. The algorithm tends toward delocalizing bonding but localizing charges, and finding the most symmetric possible solutions, as illustrated for the terminal guanidium and formamide groups in Figure 3.5.

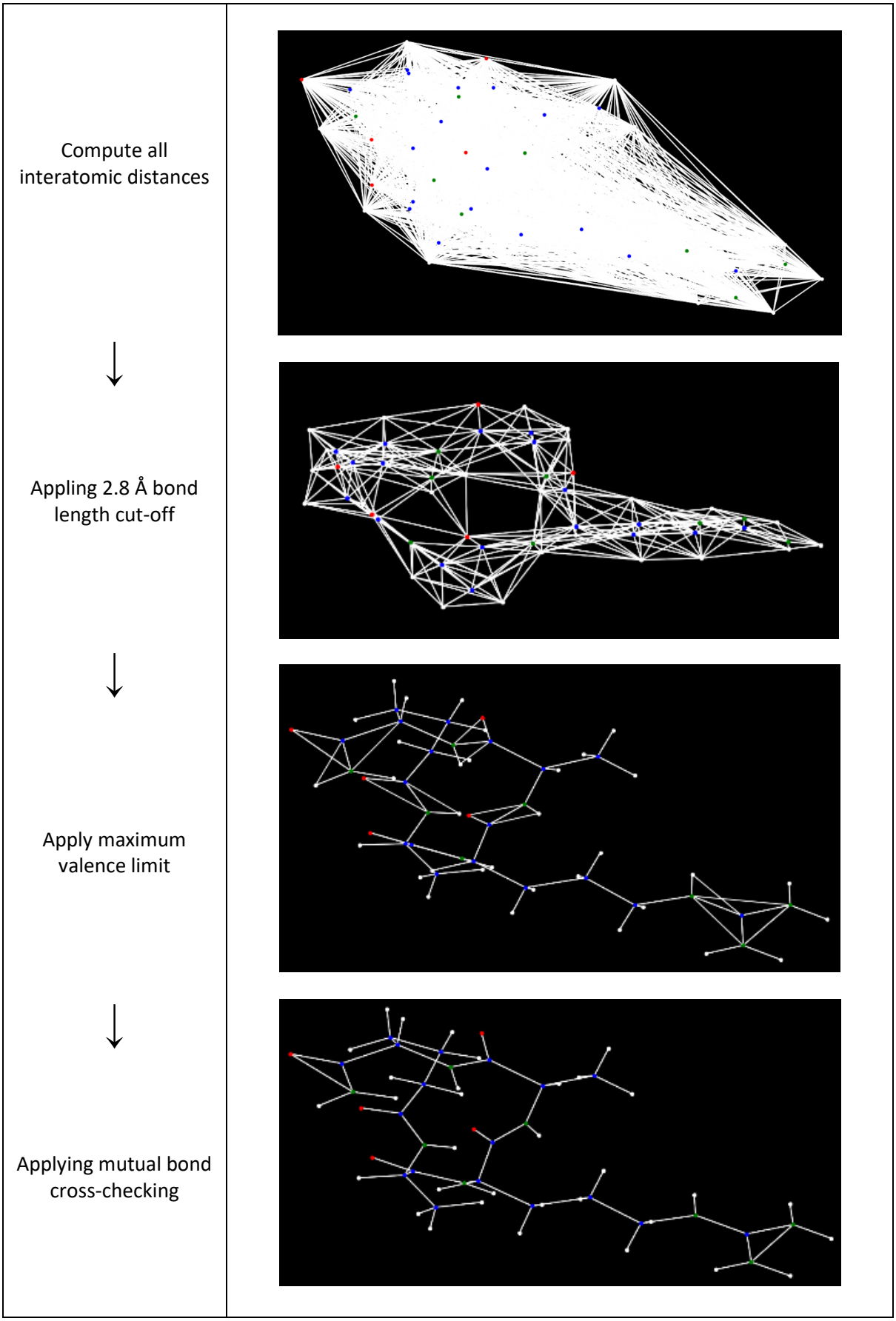


Figure 3.4. Outline of the key stages of the connectivity algorithm. At each stage, potential bonds are represented by white lines between connected atoms.

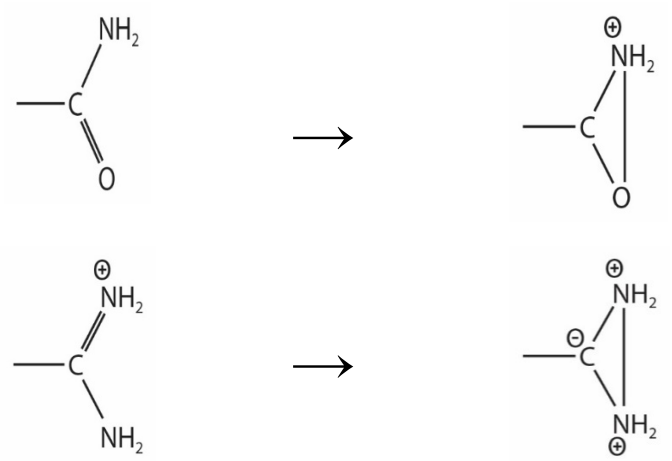
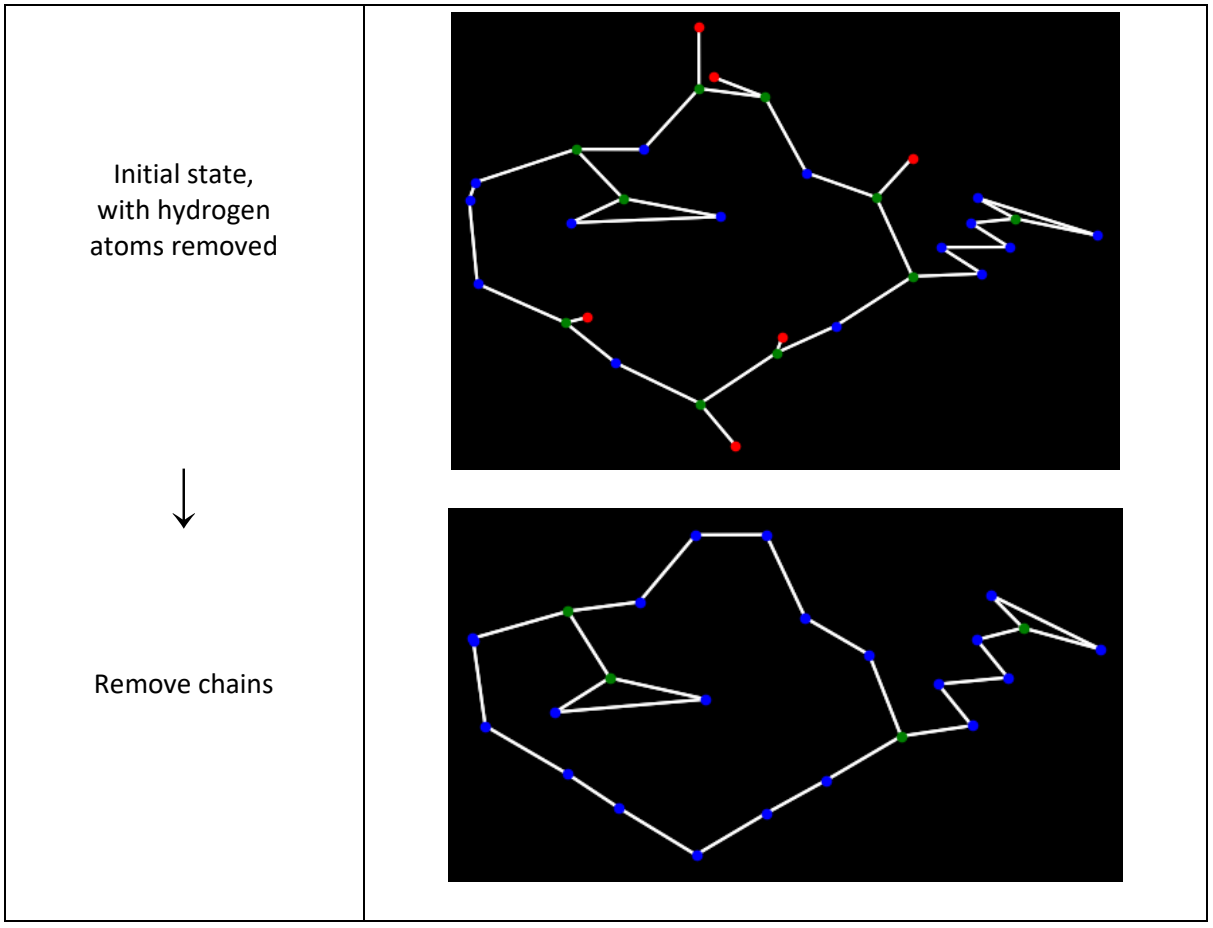


Figure 3.5. Sensible but non-conventional bonding assignments produced by the connectivity algorithm for formamide (top) and guanidinium (bottom).

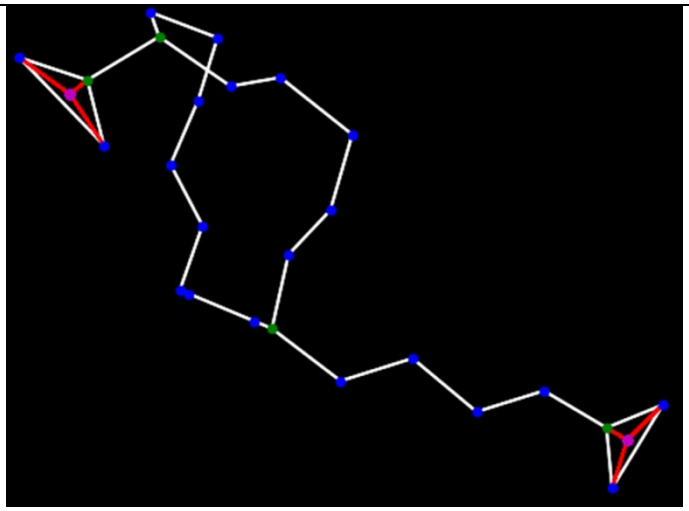
Ring finding algorithm commences

Once the connectivity algorithm has finished, the ring finding algorithm commences. This process is illustrated in Figure 3.6.



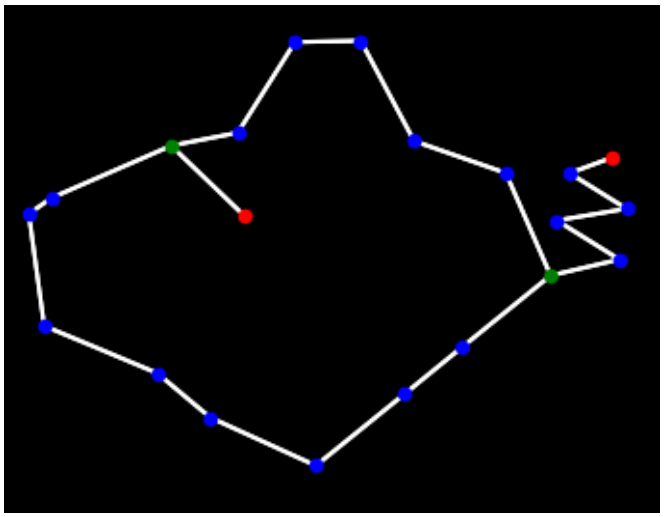
↓

Find rings,
store identities of
atoms in rings,
place central atoms



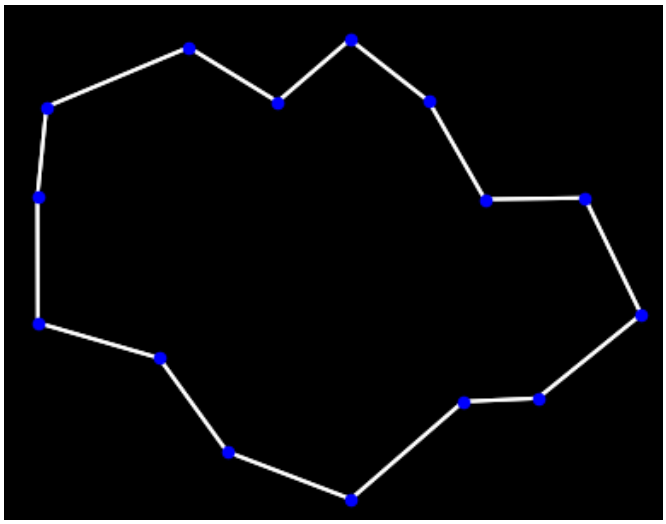
↓

Remove rings



↓

Find & remove
chain ends



↓

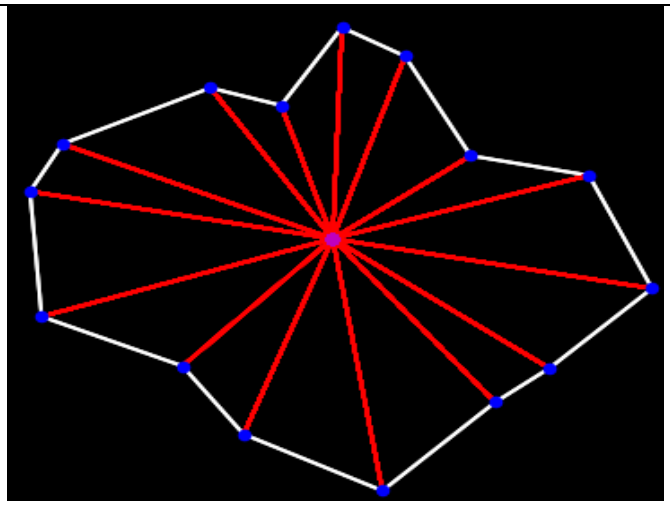
Find ring, store identities of atoms in ring, place central atom ↓ Job done	

Figure 3.6. Key stages in the ring finding algorithm. Atoms are colour-coded by their "heavy valence connectivity number", i.e. the number of non-hydrogen atoms they are connected to: Red = 1 = chain end, Blue = 2 = chain segment, Green = 3 = three-way connection point.

Complete convergence in the ring-finding code is reached once all atoms have been assigned to rings or removed as terminal chains. At this point, the "unassigned" heavy valence connectivities for all atoms have been processed down to zero.

Bonding and ring connectivities at the end of the iterative ring-finding process are illustrated in Figure 3.7.

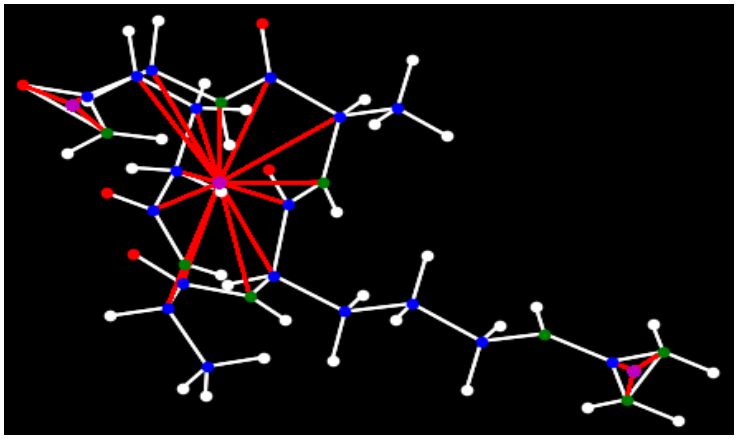


Figure 3.7: The connectivity arrangement after the ring finding algorithm has been implemented. Red connectivities are bonds with connecting external ring atoms to a central dummy placeholder.

Bond class creation and bond order and charge classification code.

Finally, some additional post-processing is carried out to convert the unconventional bonding assignments back to a chemically more conventional form, and store the final bonding and charge assignment data in an easily accessible way.

This yields the final representation illustrated in Figure 3.8.

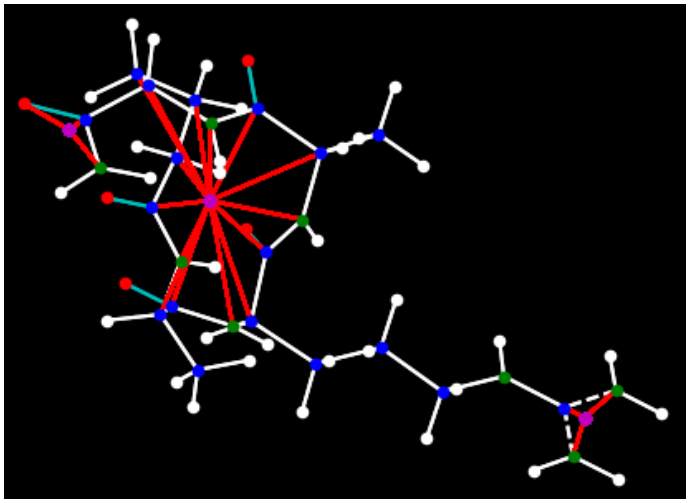


Figure 3.8. Final connectivities and bond orders. Red bonds are placeholders that denote atoms identified as being members of a ring system, and should not be considered part of the chemical structure. Dotted lines represent aromatic or resonance systems and blue lines are double bonds.

Comparing the three-dimensional structure in Figure 3.7 with the "canonical" chemical structure shown in Figure 3.3 reveals only one real chemical difference; the arginine side-chain is protonated in the NMR-derived structure but not in the canonical line drawing. However, it is well known that arginine is protonated at physiological pH, i.e. under the relevant experimental conditions.

3.3.2. Statistical analysis of outcomes

The analysis and validation flow chart is reproduced in Figure 3.8, including data on the statistical outcomes at each step.

On a superficial level, it appears that the algorithm is highly effective, processing 1488 of 1502 test molecules to complete convergence, leaving only 14 cases in which ring assignments could not be completely resolved.

However, just because bond connectivity and ring assignments can be completed doesn't necessarily mean that they are correct. Similarly, if ring assignments cannot be made, it is not necessarily true that the algorithm has encountered a chemically reasonable but topologically complex substructure – perhaps errors in the input geometry could lead to "unusual" bonding patterns that cannot be resolved.

Therefore, more detailed analysis of the results within each category is performed, as explained below and summarized in the remainder of the flow chart (Figure 3.9).

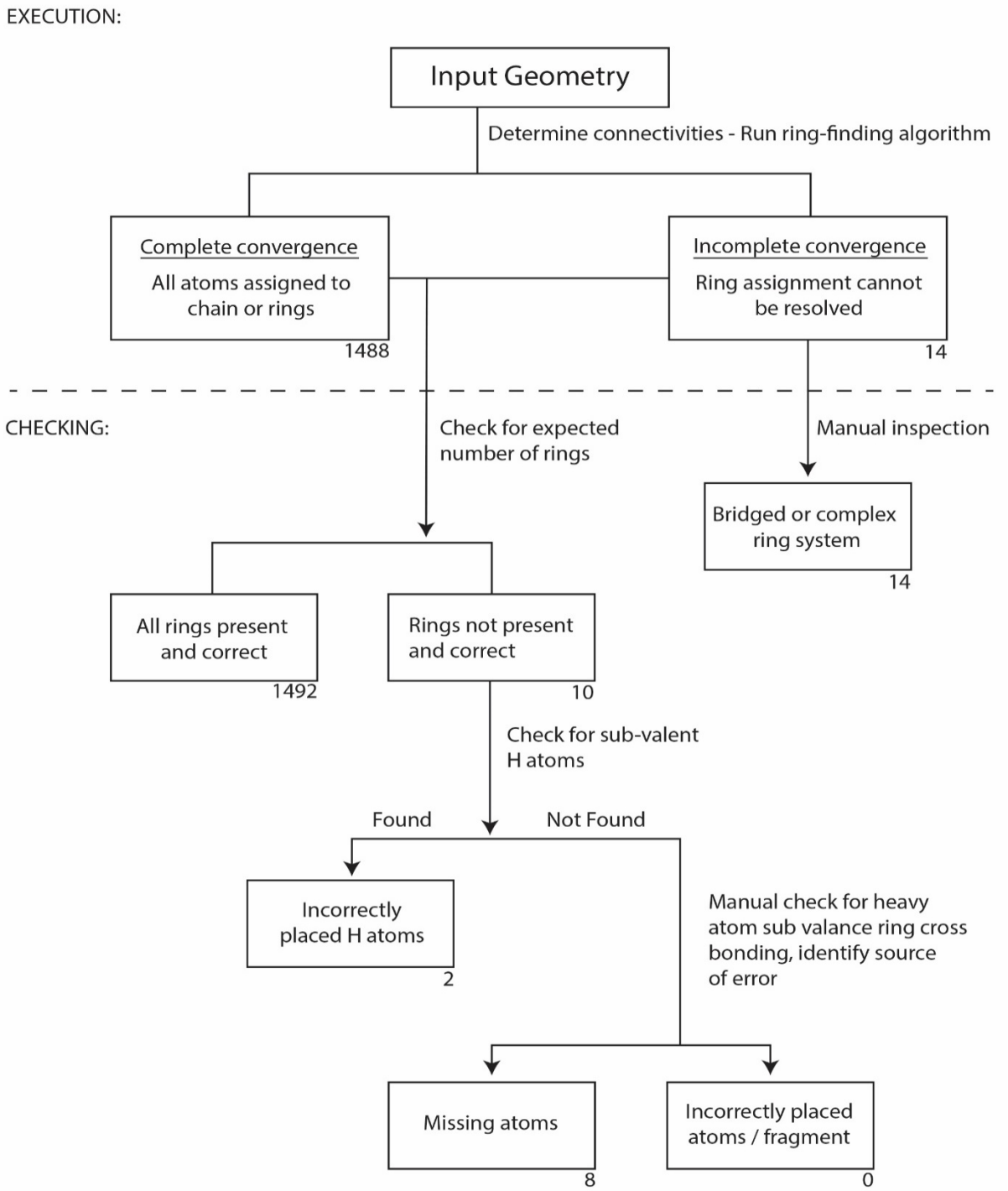


Figure 3.9. Flow chart illustrating outcomes (in boxes) of algorithm execution and checking processes (connecting lines and their labels). Branch points lead to mutually exclusive outcomes. The number of cases of each outcome is indicated by the label beneath the relevant box.

3.3.3. Incomplete convergence: topologically complex systems

The “incomplete convergence” cases are easiest to check; because there are only 14 of them, they can simply be manually inspected. Visual inspection reveals that topologically complex structures exist within all 14 molecules, and it is only these substructure fragments that cannot be resolved.

Examples of complex ring systems encountered are illustrated below.

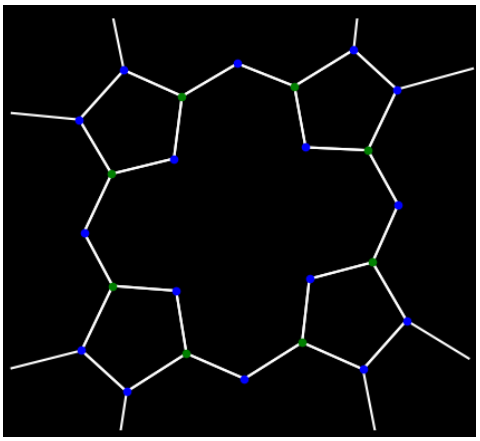


Figure 3.10. Porphyrin is an example of a topologically complex rigid macrocyclic ring system. The algorithm is incapable of processing it because the rigid macrocycle contains internal rigid rings, making it impossible to identify where a single simple ring system even begins.

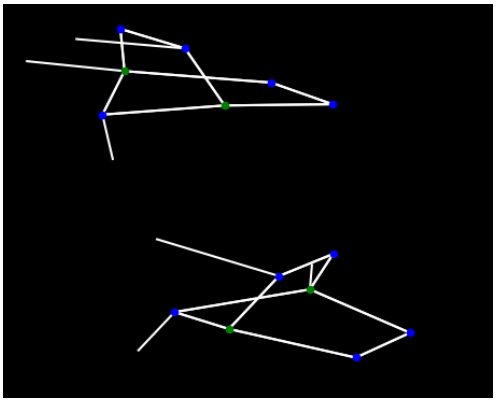


Figure 3.11. Bicyclic rings are topologically complex systems that cannot be assigned by the "simple" ring finding code. Although the bicyclic ring is comprised of two simple rings, for the processing method to accurately represent this system, it would need to identify the entire system as a single ring all at once, instead of sequentially as it currently does.

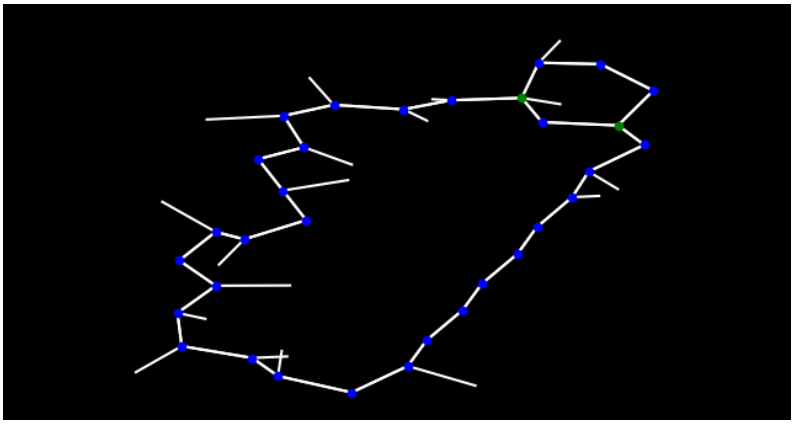


Figure 3.12. An example of a flexible complex macrocycle that contains an internal ring. Despite being structurally quite different to the rigid macrocyclic system illustrated in Figure 3.9, the ring finding code fails to process this structure for the same reason.

3.3.4. Analysis of "ring not found" errors: input errors

Having dealt with the topologically complex cases in the last section, only the cases in which the algorithm converged completely will be considered here. Despite the algorithm's overwhelming success as a whole, there were a few cases where it failed to find the rings that were expected. This section will go through examples of these failures and show that they all arise from input errors.

There are two main ways that a ring can fail to be identified in a protein or molecule:

1. Atoms are misplaced in the input: (a) hydrogen atoms, or (b) "heavy" (non-hydrogen) atoms. Examples are illustrated in Figures 3.13 and 3.14, respectively. If hydrogen atoms are misplaced, they can be counted as "bonding" to two different heavy atoms. Those atoms are then prevented from bonding to one another, hence breaking up the ring system. If heavy atoms are misplaced, bonds are not created in the first place, making ring structure identification impossible.

2. Atoms are missing from the input: (a) hydrogen atoms, or (b) entire residues. If hydrogen substituents are missing from rings, the atoms within the ring seek to fulfil their valences by creating bonds across the ring. This results in the formation of multiple rings within the ring, making it impossible to resolve the overall ring structure correctly. This situation is illustrated in Figure 3.15. The rarer case of entirely missing amino acid residues is illustrated in Figure 3.16. Either the residue has been mislabelled in the PDB file or the relevant atomic position data omitted.

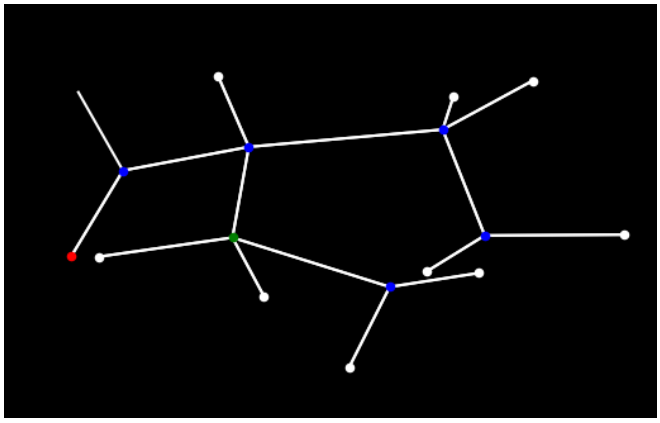


Figure 3.13. Example of an amino acid which has had its ring broken due to the misplacement of a hydrogen atom. The hydrogen atom underneath the nitrogen atom is successfully bonded to that nitrogen atom, but is also much closer to the adjacent carbon atom than it should be. This leads the carbon atom to initially believe it is a methyl group, preventing it bonding with the carbon adjacent to it and breaking up the ring.

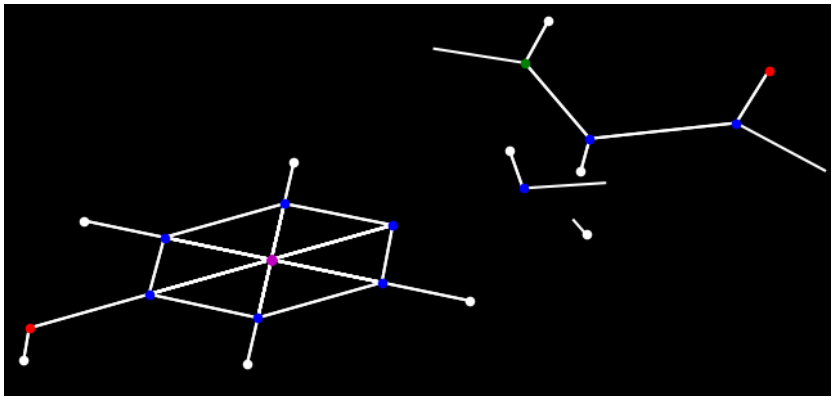


Figure 3.14. Example of a system in which a heavy atom has been misplaced. In this case, the carbon atom that connects the ring to the backbone has been misplaced, and so the connecting bonds are not formed. If this had occurred within a ring system, it would have resulted in the ring being perceived as a chain.

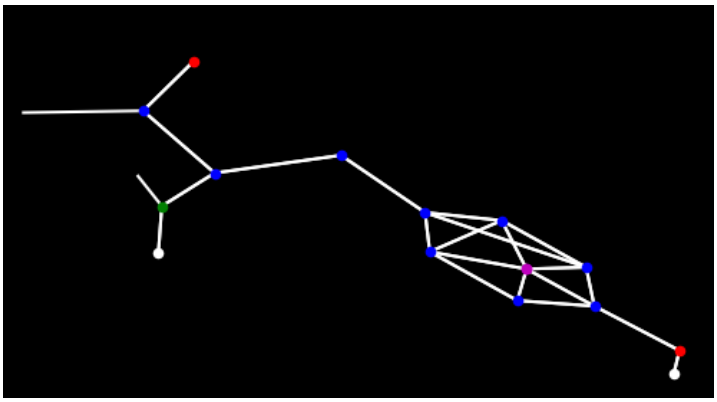


Figure 3.15: Example of amino acid with missing hydrogen atoms, that should be connected to the carbon atoms, but are not present in the original PDB file. This leads to one ring being identified but the ring has only five atoms. For an amino acid we would expect to identify a six-member ring and therefore this has been flagged this as an error.

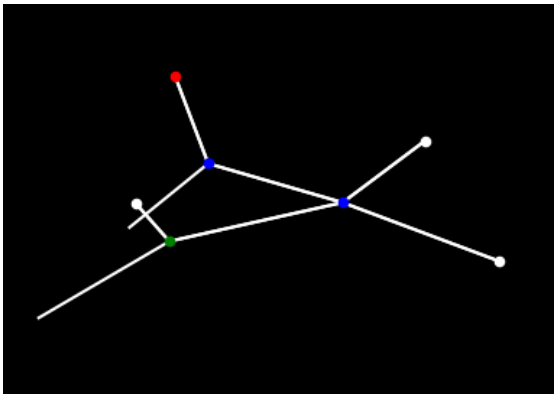


Figure 3.16. Example of amino acid in which the heavy atoms that are supposed to be part of the ring are simply not present, i.e. a ring-containing amino acid residue is replaced by glycine. It is therefore likely that either the amino acid was improperly constructed or mislabelled.

The only input error that can be automatically checked for without comparison against the PDB meta data is the case of pseudo multi valent hydrogen atoms. If a given hydrogen atom has two near-equidistant nearest neighbours (e.g. within 0.5 Å) it could easily give rise to this situation. This can be automatically detected and flagged for user attention.

3.3.5. Timing data

Having verified that the algorithm works reliably given correct input data, the next most important question becomes: how long does it take?

The most important factor in the time the algorithm takes to converge is simply the size of the protein. As shown in Figure 3.17, convergence times increase parabolically as a function of number of atoms.

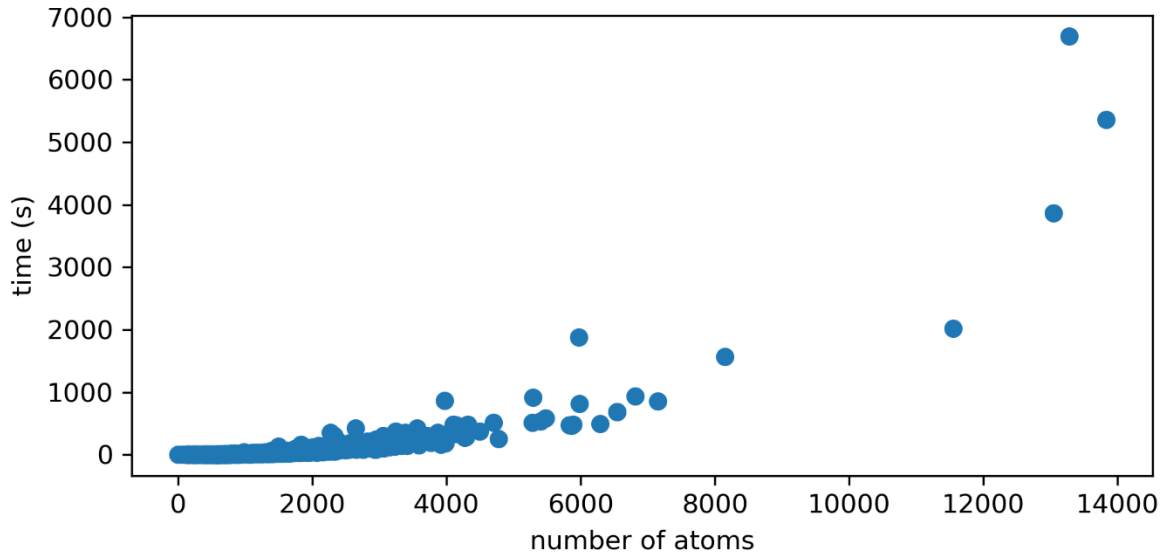


Figure 3.17. Convergence time in seconds to number of atoms in the protein.

However, there does appear to be two different parabolic curves; one steeper and one shallower. This is reflective of the two processes that occur: connectivity determination and ring-finding.

There is also a notable outlier that took around 6700 seconds to complete, taking significantly longer than the second longest time at just over 5400 seconds. Comparison of amino acid sequences reveals that this molecule does not have the most atoms, nor the most rings, nor a particularly unique or complex structure. In an attempt to understand why it takes so much longer to run than others, timing data for each iteration through the ring finding code, along with number of passes through the ring finding code are given in Table 3.1.

From this table, it can be seen that the molecule that took the longest to converge has both the longest average ring cycle time but especially the largest number of passes through the ring cycle code.

Code name	Average cycle length	Number of ring cycles	Total time
2vda	319.8691	16	6695.8850
5owi	297.0821	9	5364.3840
5id3	255.6447	6	3864.9610
1y8b	224.6467	4	2016.3630
2kr0	47.6530	31	1883.7380

Table 3.1. Ring search cycle times, number of passes through the ring searching code and total time for the five proteins with the longest overall convergence times.

3.4. Conclusions

The connectivity and ring finding algorithms presented in Chapter 2 are broadly successful in constructing accurate connectivities and identifying simple ring structures. The results generally represent chemically sensible interpretations of protein structures, obtained with almost no recourse to rule-based brute force approaches that require lookup tables for the determination of bond types.

However, in a small number cases the algorithm proceeds to complete convergence yet the structures produced are not chemically sensible, because the input data itself is not chemically sensible. The algorithm does the best job it can under the circumstances, but the “garbage in, garbage out” maxim clearly applies.

In all cases where the algorithm cannot completely resolve ring assignments (“incomplete convergence”), topologically-complex ring substructures are found to be present. This unintended feature of the algorithm - the capacity to identify complex ring systems - will be utilized in the following chapter.

4. Analysis of topologically complex natural products

4.1. Introduction

Natural products are defined as chemical substances produced by biological organisms[13, 66-67]. Natural products exhibit a variety of three-dimensional shapes that allow them to selectively bind and modulate alternative protein targets.[68-69] Natural products often exhibit high fractions of sp^3-hybridised carbons, which has been shown to increase the likelihood of successfully translating into a clinical drug candidate.[69] Therefore, natural products are an excellent source of lead compounds for drug discovery.[70-72]

Unfortunately, natural products are often difficult to obtain from the environment, even in very small quantities, and tend to be difficult to synthesize due to their structural complexity.[68] These factors generally make natural products unavailable for widespread use as drugs, although anti-cancer drug Paclitaxel (trade name Taxol) is a notable exception.[73-74] On the other hand, most mass-produced drugs on the market are simple to synthesize, but tend to be dominated by planar compounds exhibiting low fractions of sp^3 hybridised carbons.[75-76]

One way to get the best of both worlds would be to identify relatively simple three-dimensional fragments within natural products that may confer some of their specificity, and then chemically modify these fragments by attaching suitable chemical 'handles' that can then be used for later modifications, i.e. attaching additional functional groups that can be derivatized. This would enable natural-product-like molecules with specific three-dimensional shapes and chemical substituents to be systematically constructed in a synthetically efficient manner. Further, by carefully selecting the core fragment and modifications, it should be possible to build up drug molecules that are highly selective for given protein targets. Alternatively, combinatorial exploration of a range of possible cores and modifications is possible.

The goal of this chapter is to systematically identify fragments within structurally characterized natural products that are likely to have interesting three-dimensional structures. This will involve applying the ring-finding algorithm defined in Chapter 2 to the analysis of molecular structures within the Dictionary of Natural Products[77], in order to identify topologically complex ring system fragments as those left over after all chains and simple rings have been assigned. While not all

topologically complex ring systems will necessarily be three-dimensional, all rigid three-dimensional structures are likely to be topologically complex ring systems.

4.2. Methods

4.2.1 The Dictionary of Natural Products

The Dictionary of Natural Products is a database of approximately 200,000 small to medium-sized molecules derived from biological organisms whose chemical structures have been determined experimentally. Each structure is stored in structure-data file (SDF) format, containing 2D Cartesian coordinates for all non-hydrogen atoms, associated atomic connectivities, and other meta-data including accession code, molecule number, counts of functional groups of different types, and name of compound (if available).

4.2.2. Algorithmic modification

Structures reported in SDF format cannot be used as direct inputs to the combined connectivity and ring finding algorithms described in Chapter 2. The 2D coordinates do not provide enough information to obtain realistic bond lengths, and hydrogen atoms are completely missing. Therefore, it is necessary to bypass this step, and just use the supplied connectivity data from within the SDF file directly as input to the ring-finding code.

The ring-finding code largely depends only on connectivities, and so is expected to run as usual. The one exception to this is the routine that checks for and handles cross-bonding within ring systems that may be generated using by the automated connectivity detection code. However, these bonds are not present in the DNP connectivity data, which has been manually entered and verified, and so this routine should not affect the outcome.

4.2.3. Analysis of complex topologies

If the ring-finding algorithm cannot completely reduce the supplied connectivity data, i.e. assign all atoms to either chains or simple rings, then the remaining atoms are categorised as belonging to a topologically complex fragment. Information about the size of each fragment and the atoms within it

are captured and stored. Once all molecules have been analysed, all complex fragments found are sorted by size. The fragments within each size category are then resorted according to their constitutional formulae by counting numbers of carbon (C), oxygen (O), nitrogen (N) and all other (X) atoms. The overall algorithm execution and analysis process is illustrated in Figure 4.1 on following page.

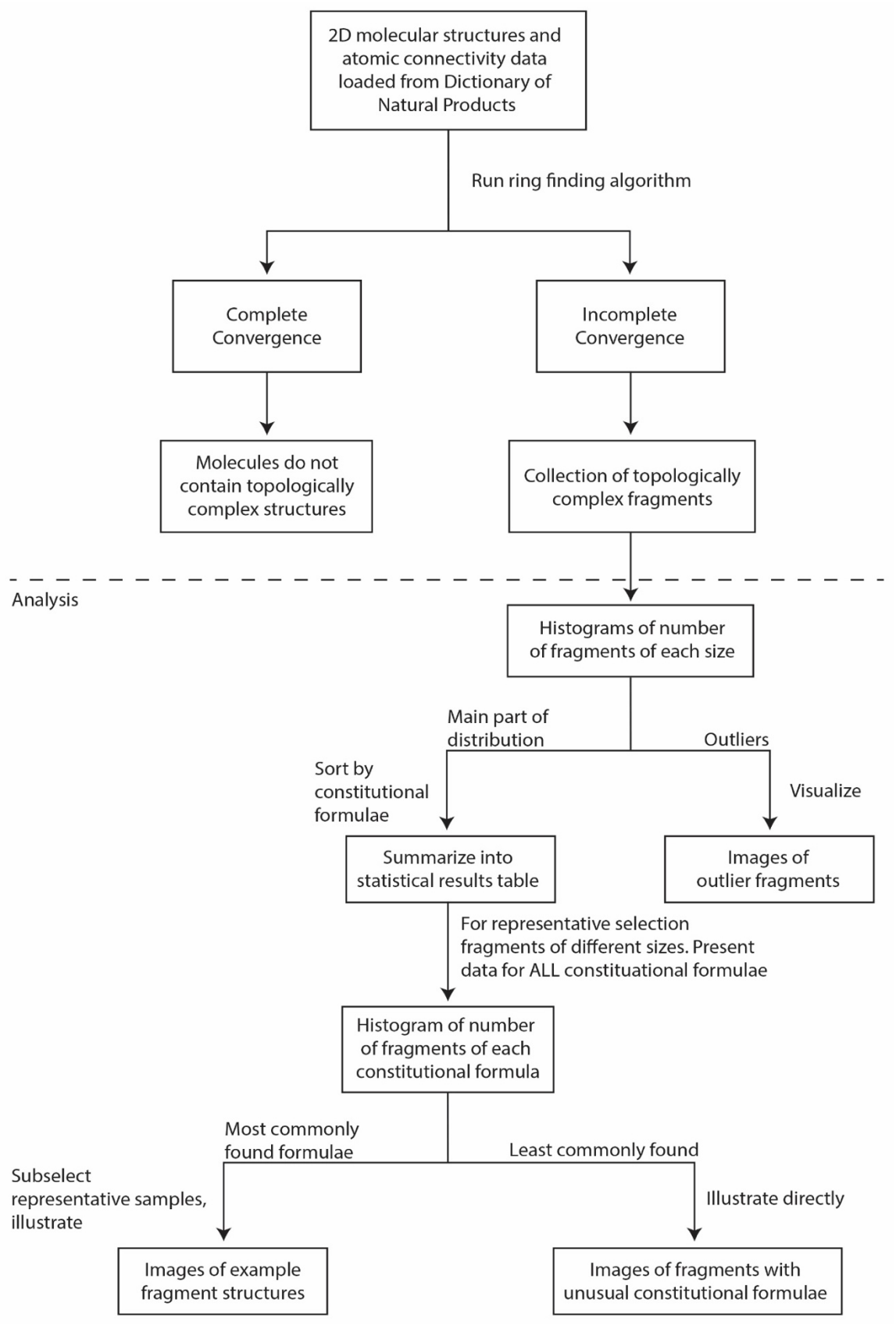


Figure 4.1: Flow chart depicting the procedure used to analyse the structures in the DNP and choose representative results to illustrate.

4.3. Results and Discussion

4.3.1. Statistical overview

Statistics on the number of molecules found to contain topologically complex fragments versus those identified as assemblies of chains and simple rings are reported in Table 4.1.

	Number	Percentage
Simple	151607	86.22
Complex	24221	13.78
Total	175828	100

Table 4.1: Breakdown of DNP composition according to: number of molecules that contain complex fragments ("Complex") vs those that do not ("Simple").

The majority of molecules within the DNP can be decomposed completely into simple fragments. Only 24,211 molecules were found to contain topologically complex fragments. The rest of this chapter is concerned with analysing the nature and composition of this fragment set.

4.3.2. Topologically complex fragment size distribution

The result of sorting all topologically complex fragments according to fragment size is illustrated in Figure 4.2.

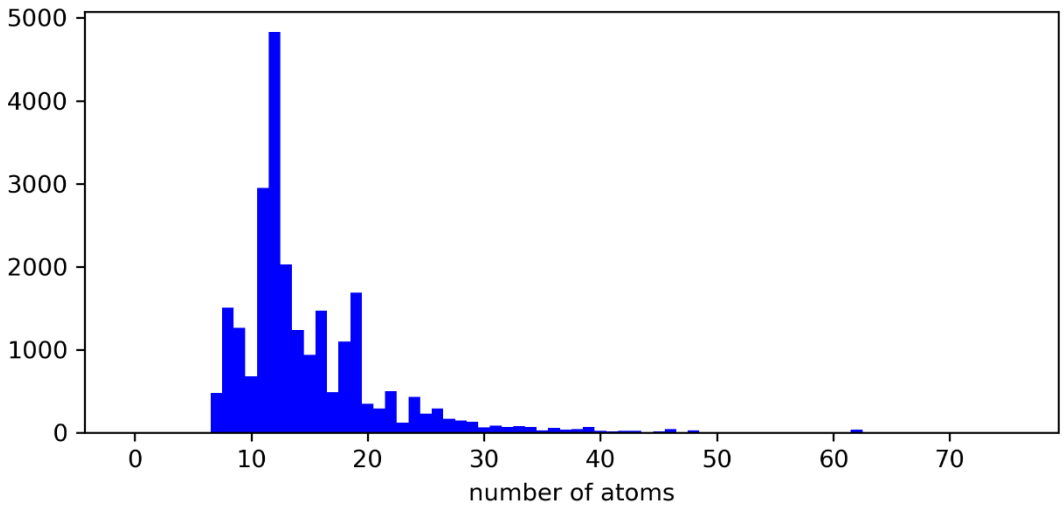


Figure 4.2: Histogram of topologically-complex fragment sizes, reporting the number of times a fragment of a specified size was found (y axis) as a function of fragment size (x axis).

The smallest identified topologically complex fragments (henceforth just referred to as “fragments” for simplicity) contain six atoms, that must consist of a topologically isolated five-membered ring with a single-atom external bridging group, e.g. five-membered ring epoxides. These occur so infrequently in the DNP (< 50/175828), however, that this category is not even visible in Figure 4.2. Topologically complex fragments containing 7 atoms are less rare but equally uninteresting; they must simply correspond to six-membered rings with single-atom bridging groups.

Fragments containing 8 and 9 atoms are far more common. However, they are unlikely to have particularly complex topologies or structures, simply due to their limited number of atoms. A detailed analysis of observed structural motifs will be investigated as a case study below.

Fragments containing 10 atoms are relatively rare (< 1000/175828), as are those containing 15, 17 and more than 20. Fragments with 10, 15 and 17 atoms are likely rare due to the bonding preferences of carbon atoms; in the same way that certain ring sizes are preferred in order to minimize steric strain, it is also likely that certain bridged-ring and fused-ring conformations are preferred.

Conversely, a wide range of connectivity and topology space is accessible to fragments with 20 atoms or more. In other words, there are many plausible ways of building larger collections of atoms into topologically complex ring systems. However, this then becomes synthetically (or biosynthetically, in this context) challenging. This is likely to the main reason for the reduced numbers of larger fragments found.

The rest of this chapter will focus on analysing differently sized fragments in more detail, categorizing all fragments of each size according to their constitution formulae, and generating histograms from which representative examples will be selected and illustrated. All illustrations use the colour scheme listed below.

Atom/bond type	**Colour**
Carbon	Blue dot
Nitrogen	Green dot
Oxygen	Red dot
Sulphur	Yellow dot
Single bond	White line
Double bond	Light blue line

Case studies are performed for the most commonly found fragment sizes (8, 9, 11, 12, 13, 14, 16, 18, 19). Fragments with 38 atoms are also analysed in more detail, as representative of larger molecules that are still within the continuous tail of the fragment size distribution. In the final case study, very large and one-off outliers are illustrated.

4.3.3. Case study: 8-atom fragments

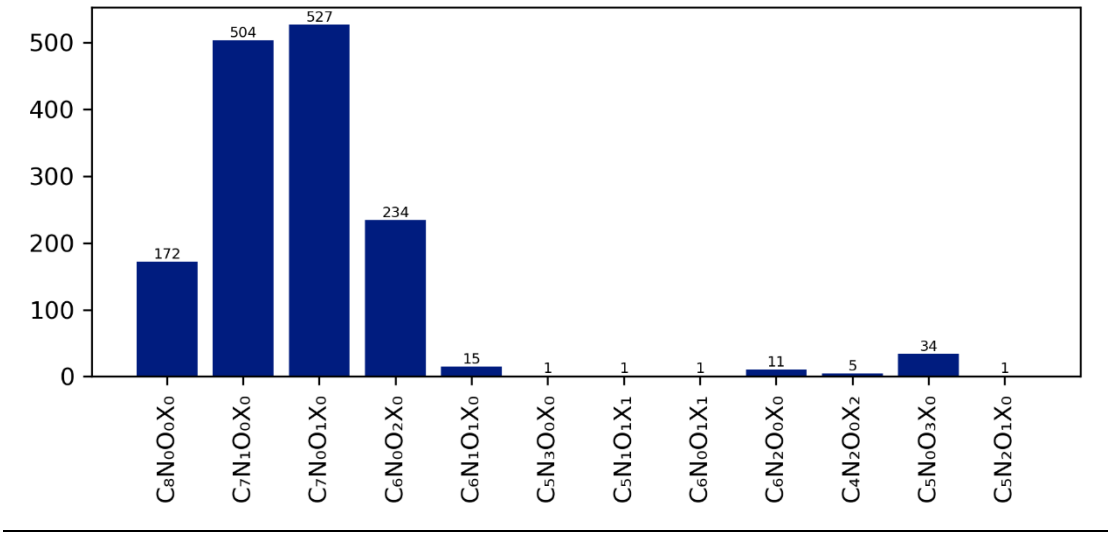


Figure 4.3: Histogram showing how frequently fragments of a given constitutional formula appear within the 8-atom fragment set. X = any atom except C, N, O.

From Figure 4.3, it is clear that most 8 atom fragments are singly-heteroatom-substituted hydrocarbons. The next most abundant are doubly-heteroatom-substituted, followed by pure hydrocarbons, then everything else.

This limited range of constitutional formulae maps to an even smaller range of topological structures; all structures sampled consisted of a six-membered ring system with two of its atoms connected via a two-atom bridge, as illustrated in Figure 4.4.

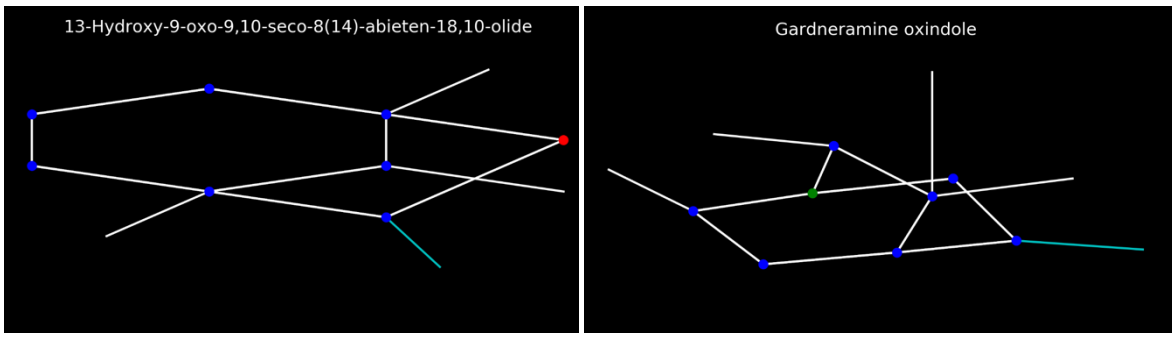


Figure 4.4. Examples of 8-atom fragments, consisting of 6-membered rings with a 2-atom (left) ortho-bridging group, and (right) para-bridging group.

Overall, the only differences between these fragments are: where the bridging system connects to the 6 membered ring, and the number and placement of the heteroatoms. This makes fragments of

this size topologically uninteresting. Therefore, in order to find more interesting three-dimensional structures, more atoms are required.

4.3.4. Case study: 9-atom fragments

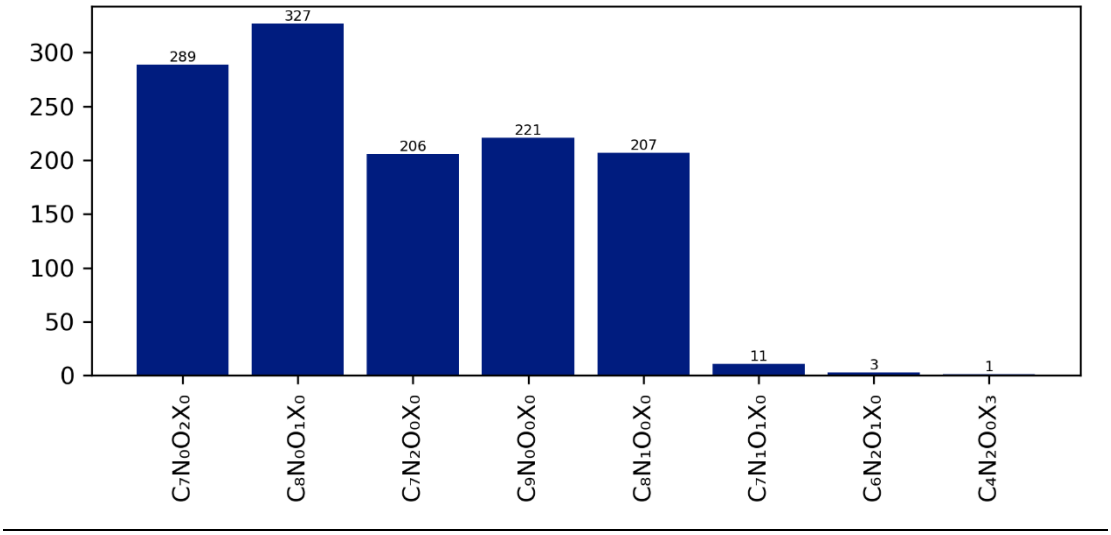


Figure 4.4: Histogram showing how frequently fragments of a given constitutional formula appear within the 9-atom fragment set. X = any atom except C, N, O.

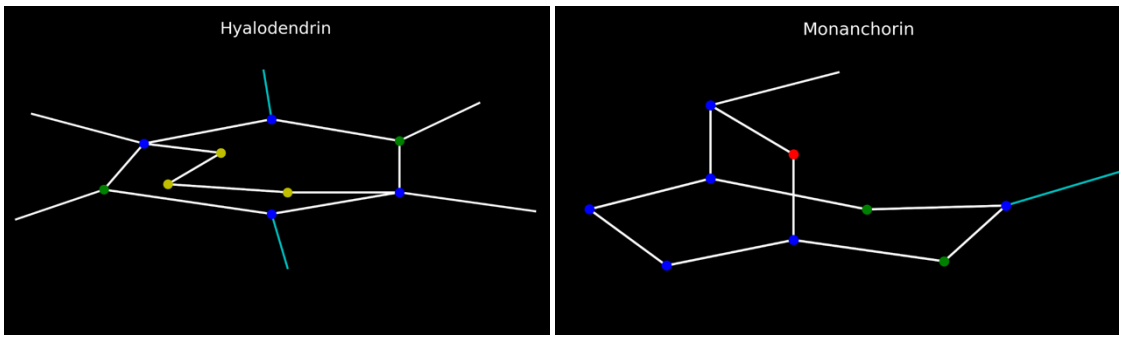


Figure 4.5. Examples of 9-atom fragments, consisting of 7-membered rings with a 2-atom bridging groups. Despite the different representations, these molecules are topologically equivalent. The structure on the left is poorly drawn and likely to have a three dimensional conformation that is better represented by the image on the right.

Like for the 8-atom fragments, the topological complexity of the 9-atom fragments is heavily restricted by the number of atoms available. All inspected examples consisted of conjoined 6- and 7-membered rings, different only in number and positioning of heteroatoms.

4.3.5. Case study: 11-atom fragments

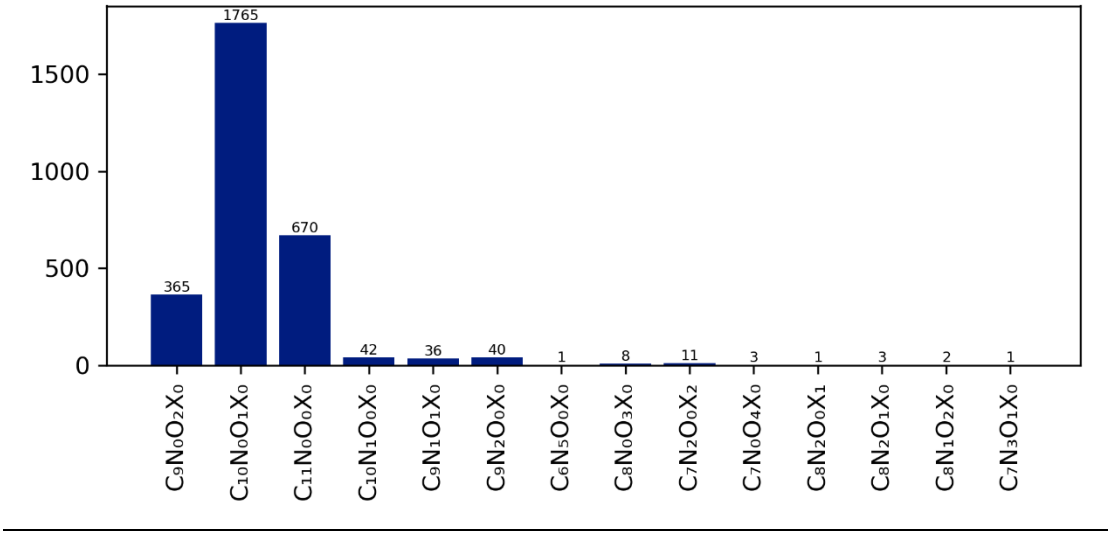


Figure 4.6: Histogram showing how frequently fragments of a given constitutional formula appear within the 11-atom fragment set. X = any atom except C, N, O.

The vast majority of 11-atom fragments are unsubstituted or oxygen-substituted hydrocarbons, with nitrogen atoms incorporated into a very small minority.

With more atoms present, more topological structure are possible. By far the most common structure is a 5-membered ring contained within an otherwise aliphatic macrocycle. However, bridged-and-fused ring systems are also possible. Examples of each of these topologies are illustrated in Figure 4.7.

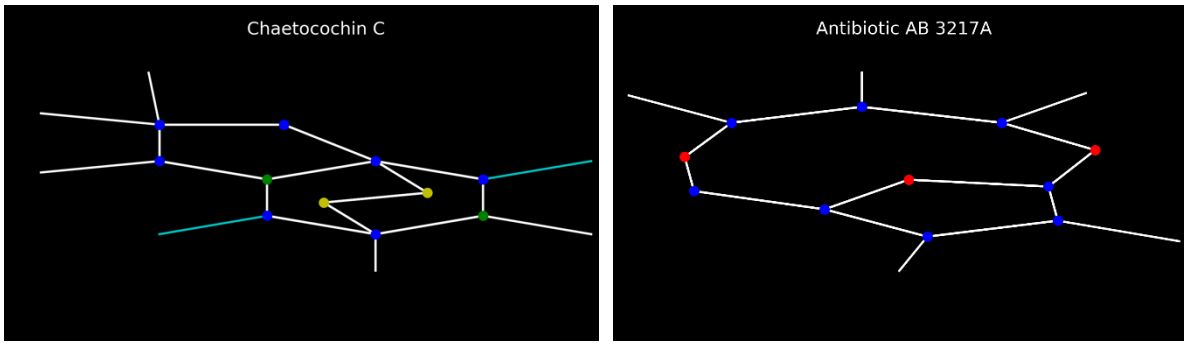


Figure 4.7: Examples of topological structures found within set of 11-atom fragments. Left: Bridged and fused ring systems. Right: Ring-within-macrocycle.

4.3.6. Case study: 12-atom fragments

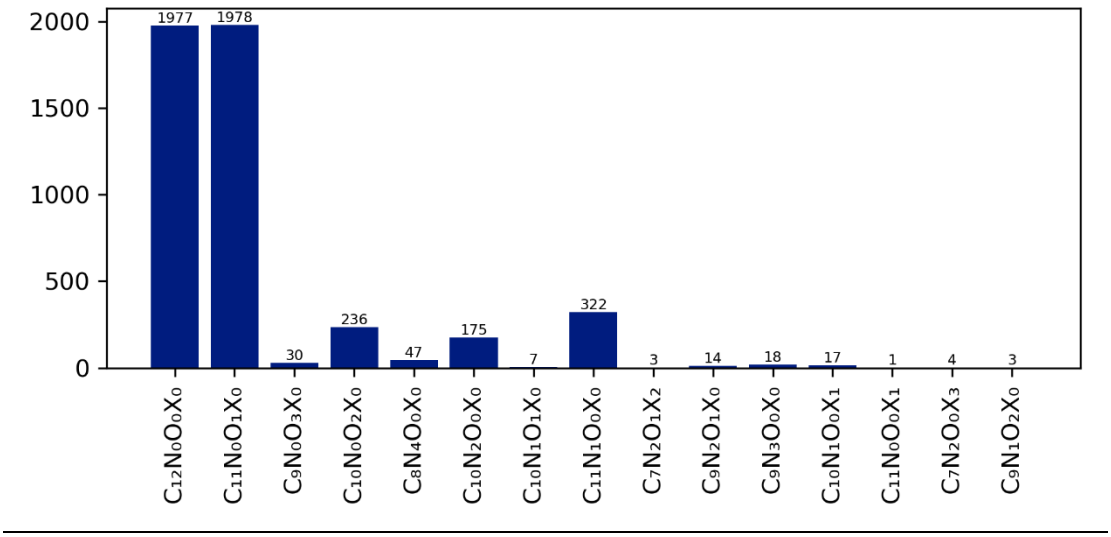


Figure 4.8: Histogram showing how frequently fragments of a given constitutional formula appear within the 12-atom fragment set. X = any atom except C, N, O.

From Figure 4.2, 12-atom fragments are the most numerous overall topologically complex structures identified within the Dictionary of Natural Products. Within the set of 12-atom fragments, Figure 4.8 shows that pure hydrocarbons and their singly heteroatom-substituted analogues occur by far the most frequently, followed by doubly heteroatom-substituted compounds.

With 12 atoms, a greater diversity of topologies is possible, and the most commonly found structural motifs – fused and bridged rings, multiply-fused rings and rings-within-macrocycles - are illustrated in Figure 4.9.

Despite the increase in topological complexity and wider range of topologies accessible, the three-dimensional structures of these molecules are not particularly interesting, as they are either mostly flat (multiply-fused rings), flat/folded (ring within aliphatic macrocycle) or consist of a flat base with a protruding bridging ring system (fused and bridged rings).

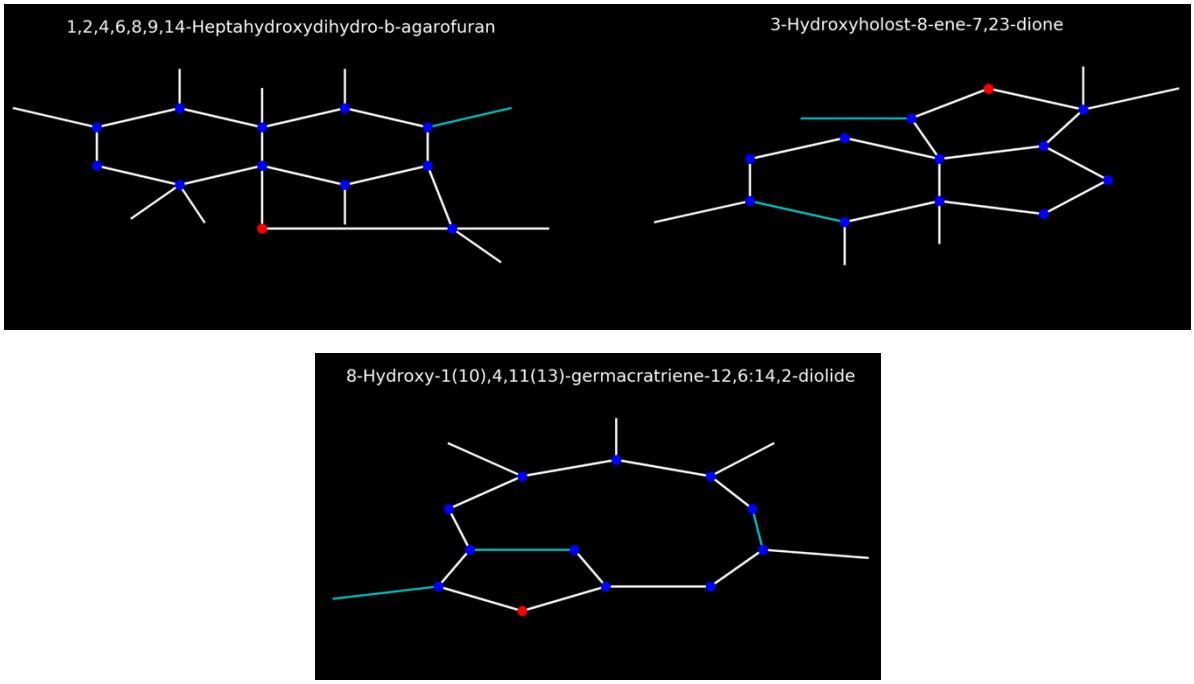


Figure 4.9: Examples of typical topologies encountered within 12-membered topologically complex ring systems. Top left: bridged-and-fused ring. Top right: Multiply-fused ring system. Bottom: Ring-within-macrocycle.

4.3.7. Case study: 13-atom fragments

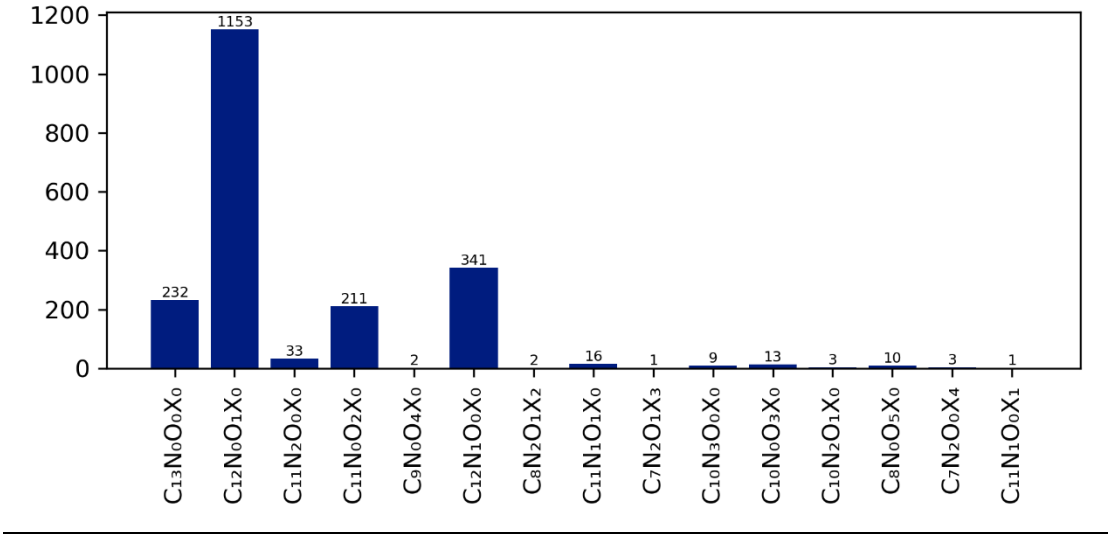


Figure 4.10: Histogram showing how frequently fragments of a given constitutional formula appear within the 13-atom fragment set. X = any atom except C, N, O.

Once again, pure hydrocarbons and singly-heteroatom-substituted hydrocarbons make up the vast majority of the fragments found.

Overall, fragments sampled from the most common constitutional formula sets tend to display structural motifs similar to those found for 12-atom systems, as show in Figure 4.11.

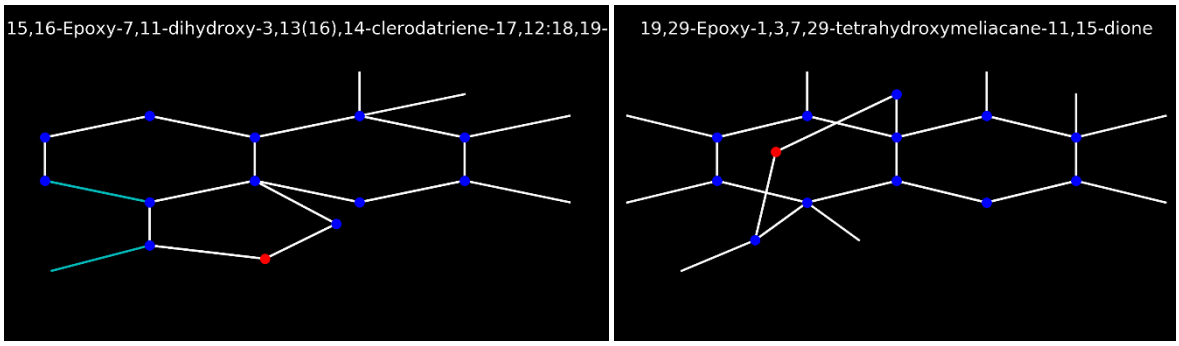


Figure 4.11: 13-atom systems with constitutional formula $C_{12}O$ display similar structural motifs to those previously identified,. Left: Multiply-fused ring system. Right: Bridged-and-fused ring system.

However, closer inspection reveals that the bridged-and-fused ring system in Figure 4.11 actually differs from those shown previously – in this case the bridge comprises three atoms rather than two, or may also be thought of as a bridge that extends out to join with a ring-substituent carbon atom.

Further, the presence of an additional atom opens up additional possibilities. Sampling from the set of fragments with constitutional formula $C_{12}N$ reveals the novel doubly-fused and bridged ring system illustrated in Figure 4.12.

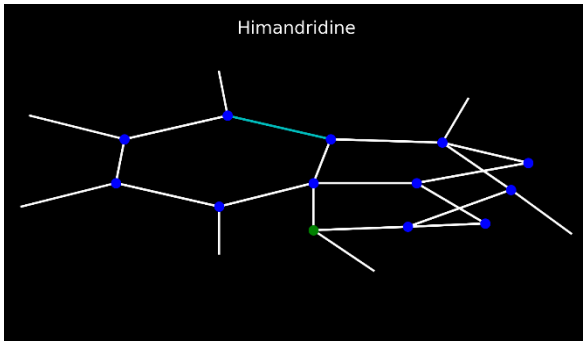


Figure 4.12: Doubly-fused and bridged ring system that appears for the first time in 13-atom fragments.

Structures sampled from the set of rare constitutional isomers $C_{11}N_1O_1$ are illustrated in Figure 4.13. One of these represents a novel three-dimensional multiply-caged structure (left), while the other is a variant of the bridged-and-fused structural motif (right). In this case, though, the bridge is not across the ring but along one side of it.

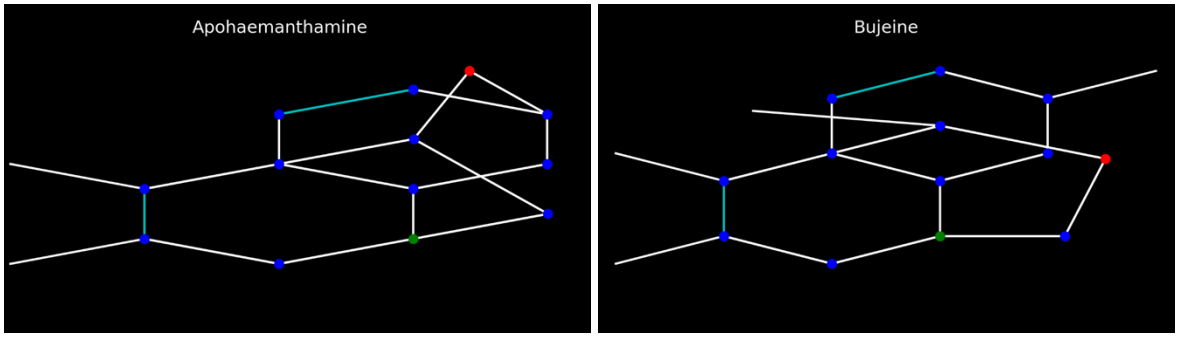


Figure 4.13: Examples of structures of fragments with constitutional formula $C_{11}N_1O_1$. Left: Multiply-bridged ring system ("cage like") and Right: Edge-fused bridging ring ("paddle-wheel like")

Overall, it appears that 13-atom fragments are the minimum size required for substantial topological complexity and diversity to emerge. This naturally translates into interesting three-dimensional structures.

4.3.8. Case study: 14-atom fragments

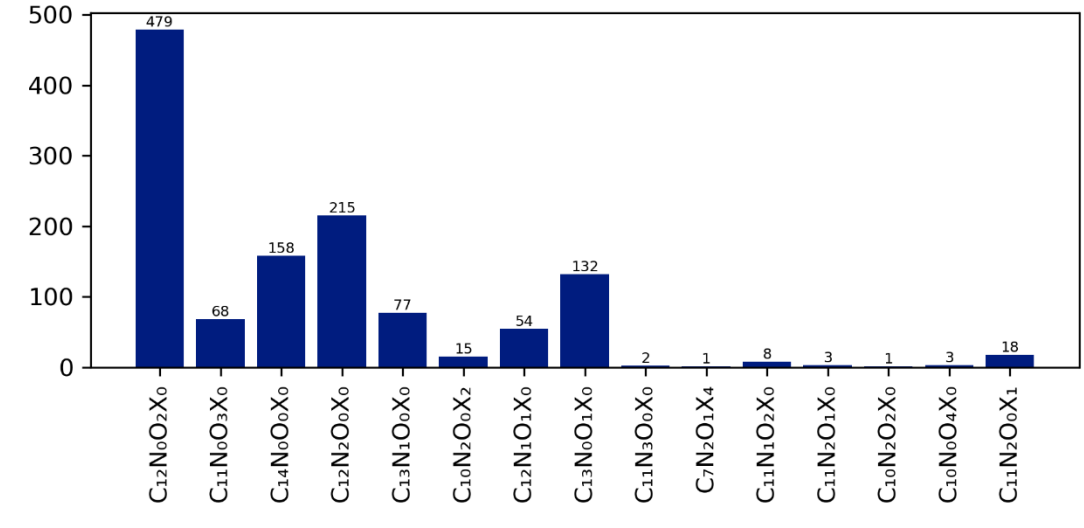


Figure 4.14: Histogram showing how frequently fragments of a given constitutional formula appear within the 14-atom fragment set. X = any atom except C, N, O.

While 12- and 13-atom fragments are among the most numerous in the DNP, Figure 4.2 shows a sharp drop once the number of atoms reaches 14. Figure 4.14 suggests that this is largely due to a drop in the number of fragments that have the same, most common constitutional formula. It is also interesting to note that singly-substituted hydrocarbons are no longer the most common fragment type. Rather, doubly-substituted hydrocarbons are more commonly found.

Sampling from the $C_{12}O_2$ fragment distribution reveals that this change in constitutional isomer distribution pattern also corresponds to a change in the topological complexity and three-dimensionality of the resultant fragments. Typical fragments are illustrated in Figure 4.15.

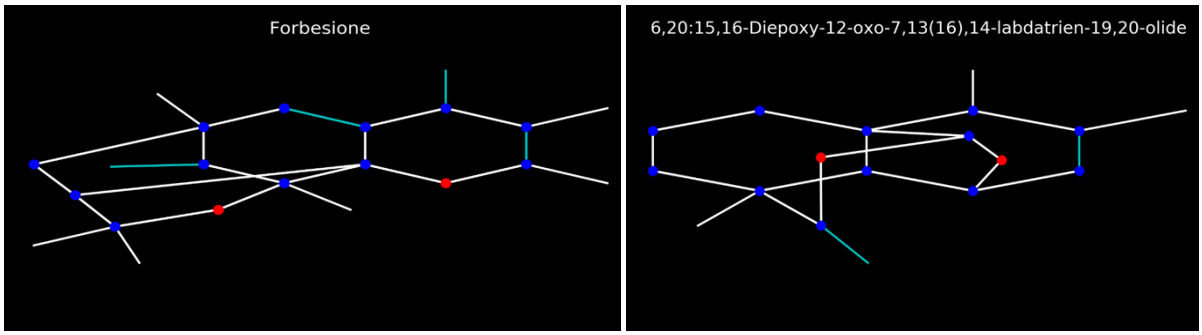


Figure 4.15: Topological connectivities for 14-atom fragments. Both fragments have novel combinations of multiply-bridged cages and planar fused ring systems.

4.3.9. Case studies: 16-, 18- and 19-atom fragments

16-, 18- and 19-atom fragments are all grouped together because they display very similar behaviour, both in terms of constitutional isomer distributions (Figure 4.16) and topological complexity of representative structures (Figure 4.17).

From Figure 4.16, singly-substituted hydrocarbons are the most commonly found, although there remains a long tail of less commonly found constitutional isomers that may also correspond to fragments with interesting and unusual connectivities and/or three-dimensional structures.

In the interests of identifying structures most commonly found in nature, fragments sampled from the sets of most commonly found constitutional isomers are illustrated in Figure 4.17. From this Figure, it appears that most structures (b-e) show a high degree of topological complexity, with multiple interconnected bridged ring systems forming overall cage-like structures. However, the relatively simple ring-in-macrocycle topology (a) is also possible in this size range, as it is for all fragments with 11 or more atoms.

To explore the topological and structural diversity accessible by including more and different heteroatoms within complex fragments, fragments sampled from uncommon constitutional isomer sets are illustrated in Figure 4.18. Both of these systems consist of a multiply-fused core, with simple

bridging across rings within the core. Structure (b) is slightly more interesting than (a), because it contains an unusual single-atom-bridged 7-membered ring system.

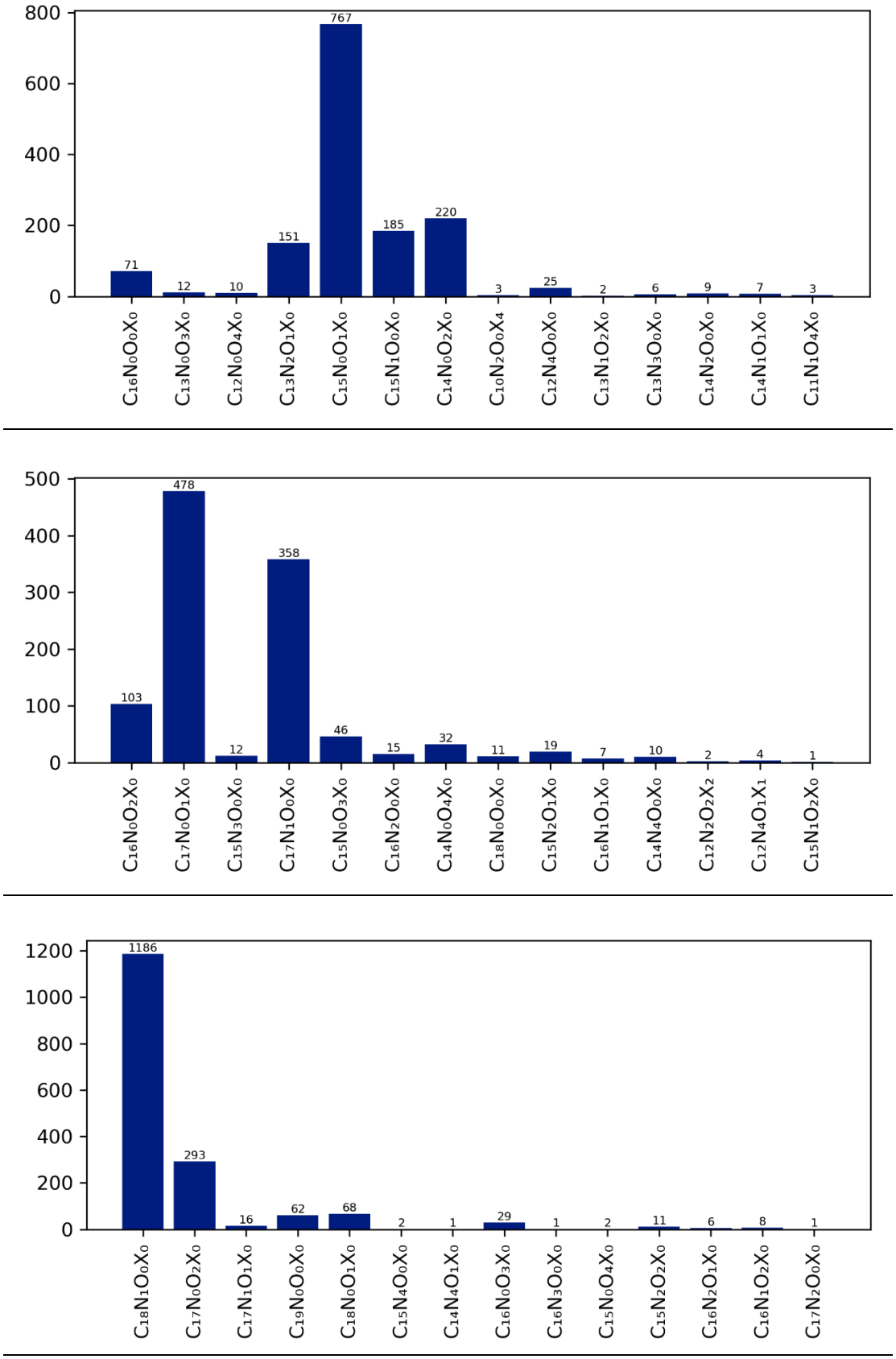
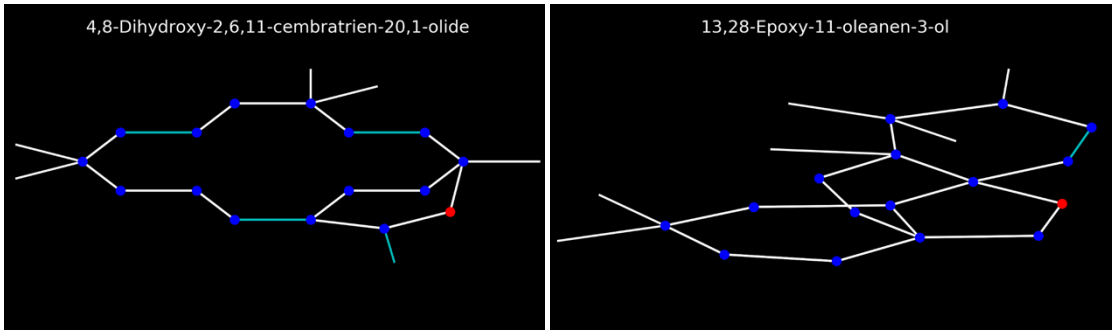
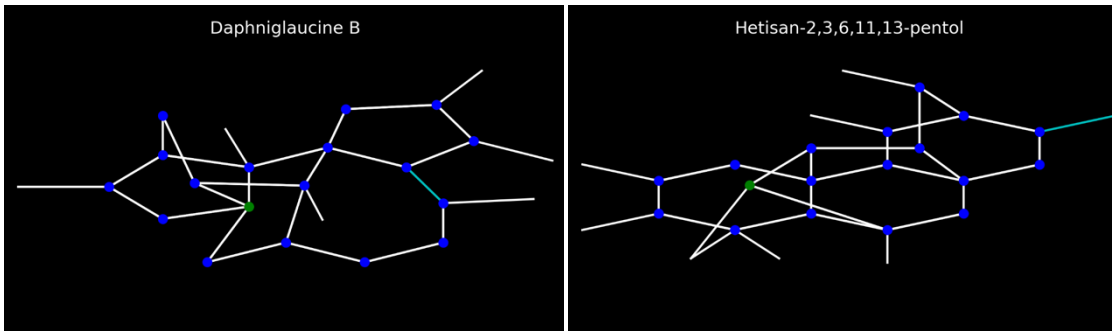


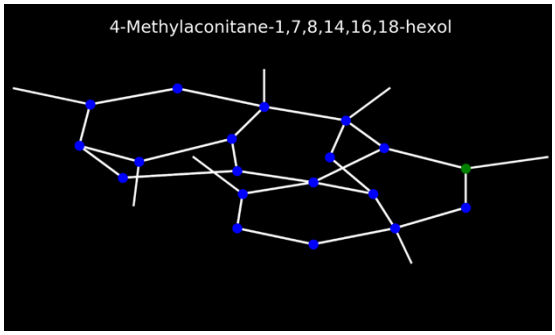
Figure 4.16: Histogram showing how frequently fragments of a given constitutional formula appear within the (top) 16-atom, (middle) 18-atom and (bottom) 19-atom fragment sets. X ≠ C, N, O.



(a) (b)

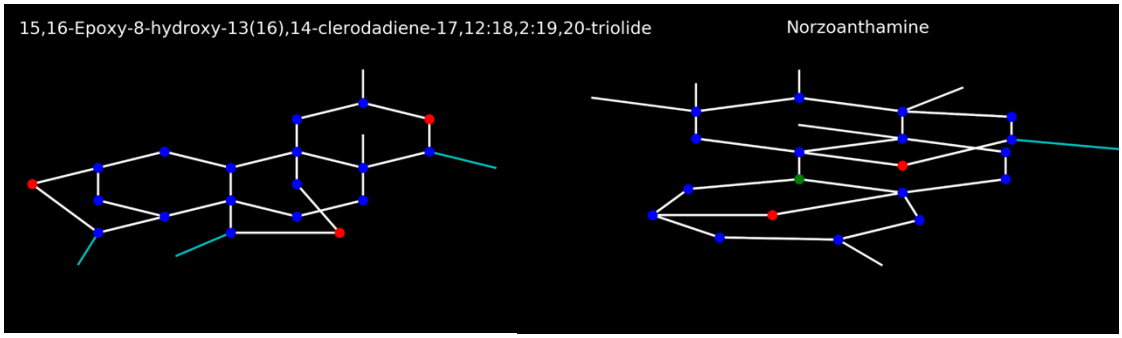


(c) (d)



(e)

Figure 4.17: Representative structures for (a,b) 16-atom fragments, (c,d) 18-atom fragments, (e) 19-atom fragments.



(a) (b)

Figure 4.18: Representative structures of uncommon constitutional isomers found amongst 19-atom fragments.

4.3.10. Case study: 38-atom fragments

Complex fragments containing 38 atoms were sampled because they are still in the main body of the fragment size distribution histogram (Figure 4.2) but are significantly larger than the most common sizes. Sample structures are illustrated in Figure 4.19.

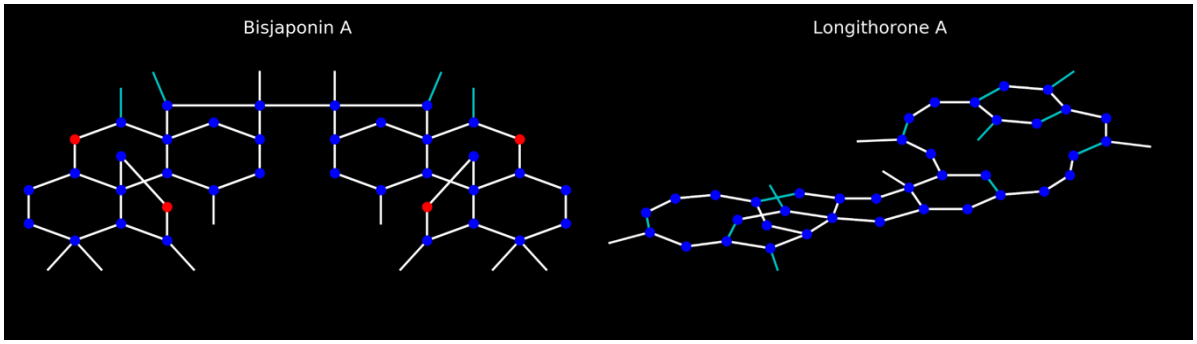


Figure 4.19: Examples of topologically-complex 38-atom fragments.

In both cases, the molecule consists of two topological complex ends connected though a simple chain or fused ring system that would otherwise be identified and removed chain-segment-by-chain-segment or ring-by-ring. However, because the molecule is capped by two complex fragments, no further ring-finding analysis is possible.

From these two examples, it can be seen that increasing the number of atoms doesn't always increase the topological complexity of the fragments, but can lead to combinations of complex structures that are otherwise simply connected.

4.3.11. Illustrated examples: Extra-large outliers

From the fragment size distribution histogram illustrated in Figure 4.2, a number of outliers with more than 50 atoms can be seen. All topologically complex fragments containing more than 70 atoms are illustrated in this section, in order to identify interesting or common topological or structural factors.

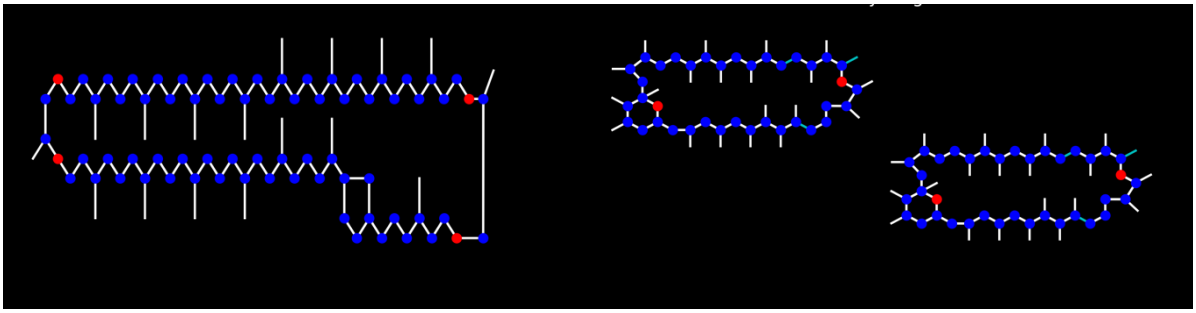


Figure 4.20: Topologically complex fragments that contain (left) 73 atoms and (right) 74 atoms

From Figure 4.20, it is clear that one easy way to generate large topologically complex fragments is to simply extend ring-in-macrocycle systems by increasing the size of the macrocycle. Also, apparently large fragments can appear as artefacts – if structures have been determined crystallographically and a unit cell with repeating structures is reported.

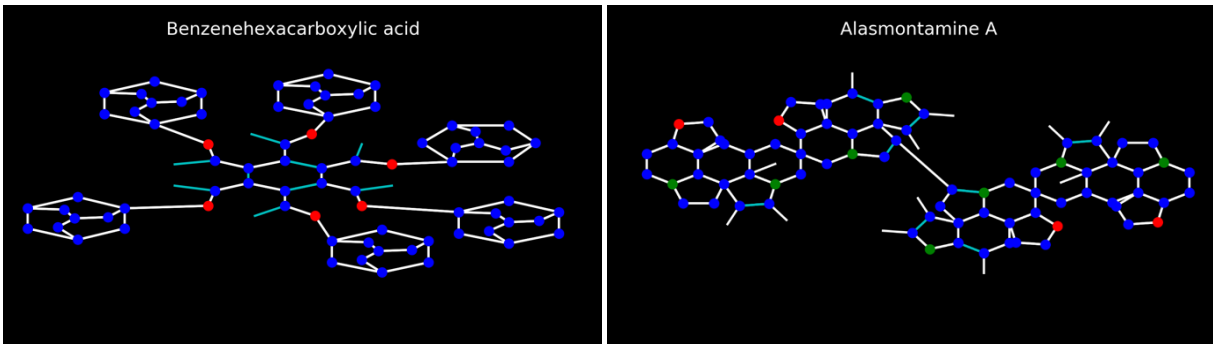


Figure 4.21: Topologically complex fragments that contain (left) 78 atoms and (right) 76 atoms

Figure 4.21 illustrates a different type of topological structure that cannot be resolved by our algorithm – cases in which all 'ends' of a molecule are capped with complex ring systems. This may be either many small complex rings protruding radially from a core, or two large complex ring systems connected linearly with one another.

In all cases, the large size of these systems simply arises from repetition of topologically and structurally uninteresting features, and so does not equate to an increase in conformational complexity, nor the emergence of interesting/useful three-dimensional structures.

4.4. Conclusions

Overall, the ring finding algorithm has successfully identified many interesting topologically complex fragments, and a series of representative examples have been identified and illustrated. Increasing fragment size allows for but does not necessarily mean that a complex fragment will have a complicated or interesting three-dimensional shape.

Neither very small nor very large fragments tend to be topologically or structurally interesting, because in both cases there are limited ways of forming the fragments. In the intermediate size regime (14-20 atoms) most of the topological and structural complexity is found.

Sorting and analysing all fragments of a given size according to their constitutional formula is useful in two ways; both for identifying fragments that occur with high frequency within the DNP but also for identifying rare fragments that may have interesting or unusual structures. However, the random sampling method applied here for visualising structures within these categories is by no means comprehensive, so the results presented herein should be read more as a list of likely outcomes than a complete structural analysis.

5.Conclusion

A robust and parameter-free algorithm has been developed to automatically detect molecular connectivities, assign bond orders and formal charges, and identify topologically simple ring systems. A topologically simple ring system is defined as a ring system in which it is possible to identify at least one unique end point, at which the ring is tethered to the rest of the molecule. This excludes bridged and caged ring systems, rings-within-rings (e.g. aromatic rings within macrocycles) and multiply-fused ring systems in which a common atom is shared between more than 2 rings.

These algorithms were validated against a test set of around 1,500 biomacromolecules whose NMR-derived structures were extracted from the RSCB protein data bank in PDB format. Additional meta-data available within the PDB files was used to determine the number of expected ring systems, and this information was used for external validation.

From this validation process, 1488 of 1502 proteins were successfully processed to complete convergence. This means that every atom in the protein has been assigned as either part of a simple ring system or part of a chain. Of the small number that failed to completely converge (14/1502), it was confirmed by visual inspection that all the atoms that could not be categorised as either part of a chain or a simple ring system were due to some form of complex topological structure existing within the protein. There was also a small subset of molecules (10/1502) in which the expected number of rings were not found, even though the algorithm had completely converged. In these cases, it was confirmed by visual inspection that this was caused by errors in the input data with either atoms being missing from the molecule or hydrogen atoms being incorrectly placed.

The ring finding algorithm that was developed and verified as outlined above was then used to search the dictionary of natural products in order to try to identify interesting three-dimensional fragments that could be used as lead compounds in the development of new drugs with specific and targeted 3D pharmacophore arrangements.

From this search, 24221/175828 molecules were found to contain topologically complex fragments. Common topologies found were multiply fused rings (flat), rings-in-macrocycles (flat or folded) and bridged rings (rigidly 3D). Small fragments (< 10 atoms) almost exclusively contained bridged ring systems. Moderately sized fragments (14-20 atoms) overall contained the widest variety of

interesting three-dimensional topologies. Finally, very large complex ring systems tended to comprise connected but repetitive complex ring systems, or were simple large rings-in-macrocycles.

6. Future Work

The connectivity and ring finding algorithms developed in this work have the ability to assign atom type, bond order and formal charges. They could therefore be used in future to generate inputs for conventional MD simulations. However, the ability to identify chains and ring systems could additionally be used to develop fragment-based force fields. The main advantages of a fragment-based approach to force field construction are improved transferability and simplified parameterization process. This would open up the possibility for running molecular dynamics simulations on wider variety of chemical systems e.g. organometallics and coordination complexes, polymers, and other nonprotein biomolecules and biopolymers.

The ability to identify topologically complex ring systems has been used to identify potential lead fragments for drug discovery. However, some of these fragments are too large and/or topologically complex to be synthesized chemically. It would therefore be useful to break these fragments down into their most fundamental topologically complex subunits. This could be achieved by systematic sub-fragmentation, which would involve systematically choosing bonds to break and then rerunning the ring finding algorithm until all of the simplest and most fundamental complex subunits have been found.

Overall, this work provides a robust platform for modelling the structures, topologies and energetics of large and/or complex macromolecules.

7. References

1. Dror, R. O.; Dirks, R. M.; Grossman, J. P.; Xu, H. F.; Shaw, D. E., Biomolecular Simulation: A Computational Microscope for Molecular Biology. In *Annual Review of Biophysics, Vol 41*, Rees, D. C., Ed. Annual Reviews: Palo Alto, 2012; Vol. 41, pp 429-452.

2. Brooks, B. R.; Brooks Iii, C. L.; Mackerell Jr, A. D.; Nilsson, L.; Petrella, R. J.; Roux, B.; Won, Y.; Archontis, G.; Bartels, C.; Boresch, S.; Caflisch, A.; Caves, L.; Cui, Q.; Dinner, A. R.; Feig, M.; Fischer, S.; Gao, J.; Hodoscek, M.; Im, W.; Kuczera, K.; Lazaridis, T.; Ma, J.; Ovchinnikov, V.; Paci, E.; Pastor, R. W.; Post, C. B.; Pu, J. Z.; Schaefer, M.; Tidor, B.; Venable, R. M.; Woodcock, H. L.; Wu, X.; Yang, W.; York, D. M.; Karplus, M., CHARMM: The biomolecular simulation program. *J. Comput. Chem.* **2009,** *30* (10), 1545-1614.

3. Oostenbrink, C.; Villa, A.; Mark, A. E.; Van Gunsteren, W. F., A biomolecular force field based on the free enthalpy of hydration and solvation: The GROMOS force-field parameter sets 53A5 and 53A6. *Journal of Computational Chemistry* **2004,** *25* (13), 1656-1676.

4. Karplus, M.; Kuriyan, J., Molecular dynamics and protein function. *Proceedings of the National Academy of Sciences of the United States of America* **2005,** *102* (19), 6679-6685.

5. Todorov, I. T.; Smith, W.; Trachenko, K.; Dove, M. T., DL_POLY_3: New dimensions in molecular dynamics simulations via massive parallelism. *Journal of Materials Chemistry* **2006,** *16* (20), 1911-1918.

6. van Gunsteren, W. F.; Burgi, R.; Peter, C.; Daura, X., The key to solving the protein-folding problem lies in an accurate description of the denatured state. *Angew. Chem.-Int. Edit.* **2001,** *40* (2), 351-355.

7. McCammon, J. A.; Gelin, B. R.; Karplus, M., Dynamics of folded proteins. *Nature* **1977,** *267* (5612), 585-590.

8. Dill, K. A.; Bromberg, S.; Yue, K.; Chan, H. S.; Ftebig, K. M.; Yee, D. P.; Thomas, P. D., Principles of protein folding — A perspective from simple exact models. *Protein Science* **1995,** *4* (4), 561-602.

9. Romero-Rivera, A.; Garcia-Borras, M.; Osuna, S., Computational tools for the evaluation of laboratory-engineered biocatalysts. *Chem. Commun.* **2017,** *53* (2), 284-297.

10. Sheldon, R. A.; Brady, D., The limits to biocatalysis: pushing the envelope. *Chem. Commun.* **2018,** *54* (48), 6088-6104.

11. Dordick, J. S., Enzymatic catalysis in monophasic organic solvents. *Enzyme and Microbial Technology* **1989,** *11* (4), 194-211.

12. Bommarius, A. S., Biocatalysis: A Status Report. In *Annual Review of Chemical and Biomolecular Engineering, Vol 6*, Prausnitz, J. M., Ed. Annual Reviews: Palo Alto, 2015; Vol. 6, pp 319-345.

13. Hardie, D. G., AMP-activated/SNF1 protein kinases: Conserved guardians of cellular energy. *Nature Reviews Molecular Cell Biology* **2007,** *8* (10), 774-785.

14. Wang, J., Electrochemical nucleic acid biosensors. *Anal. Chim. Acta* **2002,** *469* (1), 63-71.

15. Li, X.; Zhou, C. H.; Zi, Q. J.; Cao, Q. E., An electrochemical signal transduction amplification strategy for ultrasensitive detection of ascorbic acid. *J. Electroanal. Chem.* **2016,** *780*, 321-326.

16. van Gunsteren, W. F.; Bakowies, D.; Baron, R.; Chandrasekhar, I.; Christen, M.; Daura, X.; Gee, P.; Geerke, D. P.; Glattli, A.; Hunenberger, P. H.; Kastenholz, M. A.; Ostenbrink, C.; Schenk, M.; Trzesniak, D.; van der Vegt, N. F. A.; Yu, H. B., Biomolecular modeling: Goals, problems, perspectives. *Angew. Chem.-Int. Edit.* **2006,** *45* (25), 4064-4092.

17. Hardie, D. G.; Hawley, S. A., AMP-activated protein kinase: the energy charge hypothesis revisited. *Bioessays* **2001,** *23* (12), 1112-1119.

18. Borecky, J.; Vercesi, A. E., Plant uncoupling mitochondrial protein and alternative oxidase: Energy metabolism and stress. *Biosci. Rep.* **2005,** *25* (3-4), 271-286.

19. Sakurai, K.; Uezu, K.; Numata, M.; Hasegawa, T.; Li, C.; Kaneko, K.; Shinkai, S., beta-1,3-glucan polysaccharides as novel one-dimensional hosts for DNA/RNA, conjugated polymers and nanoparticles. *Chem. Commun.* **2005,** (35), 4383-4398.

20. Podgorska, M.; Kocbuch, K.; Pawelczyk, T., Recent advances in studies on biochemical and structural properties of equilibrative and concentrative nucleoside transporters. *Acta Biochim. Pol.* **2005,** *52* (4), 749-758.

21. Kuehrova, P.; Otyepka, M.; Sponer, J.; Banas, P., Are Waters around RNA More than Just a Solvent? - An Insight from Molecular Dynamics Simulations. *J. Chem. Theory Comput.* **2014,** *10* (1), 401-411.

22. Yang, B.; Zhu, Y. Y.; Wang, Y.; Chen, G. J., Interaction Identification of Zif268 and TATA(ZF) Proteins With GC-/AT-Rich DNA Sequence: A Theoretical Study. *J. Comput. Chem.* **2011,** *32* (3), 416-428.

23. Kony, D.; Damm, W.; Stoll, S.; van Gunsteren, W. F., An improved OPLS-AA force field for carbohydrates. *J. Comput. Chem.* **2002,** *23* (15), 1416-1429.

24. Schmalhorst, P. S.; Deluweit, F.; Scherrers, R.; Heisenberg, C. P.; Sikora, M., Overcoming the Limitations of the MARTINI Force Field in Simulations of Polysaccharides. *J. Chem. Theory Comput.* **2017,** *13* (10), 5039-5053.

25. Rapaport, D. C.; Rapaport, D. C. R., *The art of molecular dynamics simulation*. Cambridge university press: 2004.

26. Jorgensen, W. L.; Chandrasekhar, J.; Madura, J. D.; Impey, R. W.; Klein, M. L., Comparison of simple potential functions for simulating liquid water. *The Journal of Chemical Physics* **1983,** *79* (2), 926-935.

27. R. Bernardi, M. B., A. Bhatele, E. Bohm, R. Brunner, F. Buelens,; C. Chipot, A. D., S. Dixit, G. Fiorin, P. Freddolino, H. Fu, P. Grayson,; J. Gullingsrud, A. G., D. Hardy, C. Harrison, J. H´enin, W. Humphrey,; D. Hurwitz, A. H., N. Jain, N. Krawetz, S. Kumar, D. Kunzman,; J. Lai, C. L., R. McGreevy, C. Mei, M. Melo, M. Nelson, J. Phillips,; B. Radak, T. R., O. Sarood, A. Shinozaki, D. Tanner, D. Wells,; G. Zheng, F. Z., *NAMD User's Guide*. Version Git-2018-08-23 ed.; University of Illinois and Beckman Institute, 2018; p 254.

28. Jensen, F., *Introduction to Computational Chemistry*. Second ed.; John Wiley & Sons Ltd: 2007; p 583.

29. Lii, J. H.; Allinger, N. L., Molecular mechanics - The MM 3 force-field for hydrocarbons .3. The vendor valves potentials and Crystal data for aliphatic and aromatic hydrocarbons. *J. Am. Chem. Soc.* **1989,** *111* (23), 8576-8582.

30. Lii, J. H.; Allinger, N. L., Molecular mechanics - The MM3 force-field for hydrocarbons .2. Vibrational frequencies and thermodynamics. *J. Am. Chem. Soc.* **1989,** *111* (23), 8566-8575.

31. Allinger, N. L.; Yuh, Y. H.; Lii, J. H., Molecular mechanics - The MM3 force-field for hydrocarbons.1. *J. Am. Chem. Soc.* **1989,** *111* (23), 8551-8566.

32. Halgren, T. A., Merck molecular force field .2. MMFF94 van der Waals and electrostatic parameters for intermolecular interactions. *J. Comput. Chem.* **1996,** *17* (5-6), 520-552.

33. Halgren, T. A., Merck molecular force field .1. Basis, form, scope, parameterization, and performance of MMFF94. *J. Comput. Chem.* **1996,** *17* (5-6), 490-519.

34. D.A. Case, R. M. B., D.S. Cerutti, T.E. Cheatham, III, T.A. Darden, R.E. Duke, T.J. Giese, H. Gohlke,; A.W. Goetz, N. H., S. Izadi, P. Janowski, J. Kaus, A. Kovalenko, T.S. Lee, S. LeGrand, P. Li, C.; Lin, T. L., R. Luo, B. Madej, D. Mermelstein, K.M. Merz, G. Monard, H. Nguyen, H.T. Nguyen, I.; Omelyan, A. O., D.R. Roe, A. Roitberg, C. Sagui, C.L. Simmerling, W.M. Botello-Smith, J. Swails,; R.C. Walker, J. W., R.M. Wolf, X. Wu, L. Xiao and P.A. Kollman (2016), *AMBER 2016 Reference Manual*. University of California, San Francisco, 2016.

35. Banks, J. L.; Beard, H. S.; Cao, Y.; Cho, A. E.; Damm, W.; Farid, R.; Felts, A. K.; Halgren, T. A.; Mainz, D. T.; Maple, J. R.; Murphy, R.; Philipp, D. M.; Repasky, M. P.; Zhang, L. Y.; Berne, B. J.; Friesner, R. A.; Gallicchio, E.; Levy, R. M., Integrated Modeling Program, Applied Chemical Theory (IMPACT). *Journal of Computational Chemistry* **2005,** *26* (16), 1752-1780.

36. Vanommeslaeghe, K.; Raman, E. P.; MacKerell, A. D., Automation of the CHARMM General Force Field (CGenFF) II: Assignment of Bonded Parameters and Partial Atomic Charges. *Journal of Chemical Information and Modeling* **2012,** *52* (12), 3155-3168.

37. Vanommeslaeghe, K.; MacKerell Jr, A. D., Automation of the CHARMM general force field (CGenFF) I: Bond perception and atom typing. *Journal of Chemical Information and Modeling* **2012,** *52* (12), 3144-3154.

38. Malde, A. K.; Zuo, L.; Breeze, M.; Stroet, M.; Poger, D.; Nair, P. C.; Oostenbrink, C.; Mark, A. E., An Automated force field Topology Builder (ATB) and repository: Version 1.0. *J. Chem. Theory Comput.* **2011,** *7* (12), 4026-4037.

39. Wang, J.; Wang, W.; Kollman, P. A.; Case, D. A., Automatic atom type and bond type perception in molecular mechanical calculations. *Journal of Molecular Graphics and Modelling* **2006,** *25* (2), 247-260.

40. Homeyer, N.; Horn, A. H. C.; Lanig, H.; Sticht, H., AMBER force-field parameters for phosphorylated amino acids in different protonation states: Phosphoserine, phosphothreonine, phosphotyrosine, and phosphohistidine. *Journal of Molecular Modeling* **2006,** *12* (3), 281-289.

41. Wang, J.; Wolf, R. M.; Caldwell, J. W.; Kollman, P. A.; Case, D. A., Development and testing of a general Amber force field. *J. Comput. Chem.* **2004,** *25* (9), 1157-1174.

42. Zhang, Q.; Zhang, W.; Li, Y. Y.; Wang, J. M.; Zhang, L. L.; Hou, T. J., A rule-based algorithm for automatic bond type perception. *J. Cheminformatics* **2012,** *4*, 10.

43. Maier, J. A.; Martinez, C.; Kasavajhala, K.; Wickstrom, L.; Hauser, K. E.; Simmerling, C., ff14SB: Improving the Accuracy of Protein Side Chain and Backbone Parameters from ff99SB. *J. Chem. Theory Comput.* **2015,** *11* (8), 3696-3713.

44. Galindo-Murillo, R.; Robertson, J. C.; Zgarbova, M.; Sponer, J.; Otyepka, M.; Jurecka, P.; Cheatham, T. E., Assessing the Current State of Amber Force Field Modifications for DNA. *J. Chem. Theory Comput.* **2016,** *12* (8), 4114-4127.

45. Zgarbova, M.; Sponer, J.; Otyepka, M.; Cheatham, T. E.; Galindo-Murillo, R.; Jurecka, P., Refinement of the Sugar-Phosphate Backbone Torsion Beta for AMBER Force Fields Improves the Description of Z- and B-DNA. *J. Chem. Theory Comput.* **2015,** *11* (12), 5723-5736.

46. Banas, P.; Hollas, D.; Zgarbova, M.; Jurecka, P.; Orozco, M.; Cheatham, T. E.; Sponer, J.; Otyepka, M., Performance of Molecular Mechanics Force Fields for RNA Simulations: Stability of UUCG and GNRA Hairpins. *J. Chem. Theory Comput.* **2010,** *6* (12), 3836-3849.

47. Zgarbova, M.; Otyepka, M.; Sponer, J.; Mladek, A.; Banas, P.; Cheatham, T. E.; Jurecka, P., Refinement of the Cornell et al. Nucleic Acids Force Field Based on Reference Quantum Chemical Calculations of Glycosidic Torsion Profiles. *J. Chem. Theory Comput.* **2011,** *7* (9), 2886-2902.

48. Kirschner, K. N.; Yongye, A. B.; Tschampel, S. M.; Gonzalez-Outeirino, J.; Daniels, C. R.; Foley, B. L.; Woods, R. J., GLYCAM06: A generalizable Biomolecular force field. Carbohydrates. *J. Comput. Chem.* **2008,** *29* (4), 622-655.

49. Dickson, C. J.; Madej, B. D.; Skjevik, A. A.; Betz, R. M.; Teigen, K.; Gould, I. R.; Walker, R. C., Lipid14: The Amber Lipid Force Field. *J. Chem. Theory Comput.* **2014,** *10* (2), 865-879.

50. Jorgensen, W. L.; Chandrasekhar, J.; Madura, J. D.; Impey, R. W.; Klein, M. L., COMPARISON OF SIMPLE POTENTIAL FUNCTIONS FOR SIMULATING LIQUID WATER. *J. Chem. Phys.* **1983,** *79* (2), 926-935.

51. Horn, H. W.; Swope, W. C.; Pitera, J. W., Characterization of the TIP4P-Ew water model: Vapor pressure and boiling point. *J. Chem. Phys.* **2005,** *123* (19), 12.

52. Izadi, S.; Anandakrishnan, R.; Onufriev, A. V., Building Water Models: A Different Approach. *J. Phys. Chem. Lett.* **2014,** *5* (21), 3863-3871.

53. Pérez, A.; Marchán, I.; Svozil, D.; Sponer, J.; Cheatham Iii, T. E.; Laughton, C. A.; Orozco, M., Refinement of the AMBER force field for nucleic acids: Improving the description of α/γ conformers. *Biophysical Journal* **2007,** *92* (11), 3817-3829.

54. Sun, H.; Mumby, S. J.; Maple, J. R.; Hagler, A. T., An ab Initio CFF93 All-Atom Force Field for Polycarbonates. *J. Am. Chem. Soc.* **1994,** *116* (7), 2978-2987.

55. Koziara, K. B.; Stroet, M.; Malde, A. K.; Mark, A. E., Testing and validation of the Automated Topology Builder (ATB) version 2.0: Prediction of hydration free enthalpies. *Journal of Computer-Aided Molecular Design* **2014,** *28* (3), 221-233.

56. Stroet, M.; Caron, B.; Visscher, K. M.; Geerke, D. P.; Malde, A. K.; Mark, A. E., Automated Topology Builder Version 3.0: Prediction of Solvation Free Enthalpies in Water and Hexane. *J. Chem. Theory Comput.* **2018,** *14* (11), 5834-5845.

57. van Gunsteren, W. F.; Berendsen, H. J. C., Computer Simulation of Molecular Dynamics: Methodology, Applications, and Perspectives in Chemistry. *Angewandte Chemie International Edition in English* **1990,** *29* (9), 992-1023.

58. Comba, P.; Remenyi, R., Inorganic and bioinorganic molecular mechanics modeling - The problem of the force field parameterization. *Coordination Chemistry Reviews* **2003,** *238-239*, 9-20.

59. H.M. Berman, J. W., Z. Feng, G. Gilliland, T.N. Bhat, H. Weissig, I.N. Shindyalov, P.E. Bourne., The Protein Data Bank Nucleic Acids Research. (2000) Vol. 28:, pp 235-242.

60. Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T. N.; Weissig, H.; Shindyalov, I. N.; Bourne, P. E., The Protein Data Bank. *Nucleic Acids Research* **2000,** *28* (1), 235-242.

61. Berman, H.; Henrick, K.; Nakamura, H.; Markley, J. L., The worldwide Protein Data Bank (wwPDB): Ensuring a single, uniform archive of PDB data. *Nucleic Acids Research* **2007,** *35* (SUPPL. 1), D301-D303.

62. Muller, P., Practical suggestions for better crystal structures. *Crystallogr. Rev* **2009,** *15* (1), 57-83.

63. Billeter, M.; Wagner, G.; Wüthrich, K., Solution NMR structure determination of proteins revisited. *Journal of Biomolecular NMR* **2008,** *42* (3), 155-158.

64. Koradi, R.; Billeter, M.; Wüthrich, K., MOLMOL: A program for display and analysis of macromolecular structures. *Journal of Molecular Graphics* **1996,** *14* (1), 51-55.

65. Gronenborn, A. M., Determination of Three-Dimensional Structures of Proteins and Nucleic Acids in Solution by Nuclear Magnetic Resonance Spectroscop AU - Clore, G. Marius. *Critical Reviews in Biochemistry and Molecular Biology* **1989,** *24* (5), 479-564.

66. All natural. *Nature Chemical Biology* **2007,** *3*, 351.

67. Davison, E. K.; Brimble, M. A., Natural product derived privileged scaffolds in drug discovery. *Current Opinion in Chemical Biology* **2019,** *52*, 1-8.

68. Van Hattum, H.; Waldmann, H., Biology-oriented synthesis: Harnessing the power of evolution. *J. Am. Chem. Soc.* **2014,** *136* (34), 11853-11859.

69. Lovering, F.; Bikker, J.; Humblet, C., Escape from flatland: Increasing saturation as an approach to improving clinical success. *Journal of Medicinal Chemistry* **2009,** *52* (21), 6752-6756.

70. Borris, R. P., Natural products research: Perspectives from a major pharmaceutical company. *J. Ethnopharmacol.* **1996,** *51* (1-3), 29-38.

71. Daglia, M., Polyphenols as antimicrobial agents. *Curr. Opin. Biotechnol.* **2012,** *23* (2), 174-181.

72. Newman, D. J.; Cragg, G. M., Natural Products as Sources of New Drugs from 1981 to 2014. *Journal of Natural Products* **2016,** *79* (3), 629-661.

73. Peltier, S.; Oger, J. M.; Lagarce, F.; Couet, W.; Benoit, J. P., Enhanced oral paclitaxel bioavailability after administration of paclitaxel-loaded lipid nanocapsules. *Pharm. Res.* **2006,** *23* (6), 1243-1250.

74. Sandler, A.; Gray, R.; Perry, M. C.; Brahmer, J.; Schiller, J. H.; Dowlati, A.; Lilenbaum, R.; Johnson, D. H., Paclitaxel-carboplatin alone or with bevacizumab for non-small-cell lung cancer. *N. Engl. J. Med.* **2006,** *355* (24), 2542-2550.

75. Erlanson, D. A.; Fesik, S. W.; Hubbard, R. E.; Jahnke, W.; Jhoti, H., Twenty years on: the impact of fragments on drug discovery. *Nat. Rev. Drug Discov.* **2016,** *15* (9), 605-619.

76. Morley, A. D.; Pugliese, A.; Birchall, K.; Bower, J.; Brennan, P.; Brown, N.; Chapman, T.; Drysdale, M.; Gilbert, I. H.; Hoelder, S.; Jordan, A.; Ley, S. V.; Merritt, A.; Miller, D.; Swarbrick, M. E.; Wyatt, P. G., Fragment-based hit identification: thinking in 3D. *Drug Discov. Today* **2013,** *18* (23-24), 1221-1227.

77. Buckingham, J., *Dictionary of Natural Products, Supplement 4*. CRC press: 1997; Vol. 11.